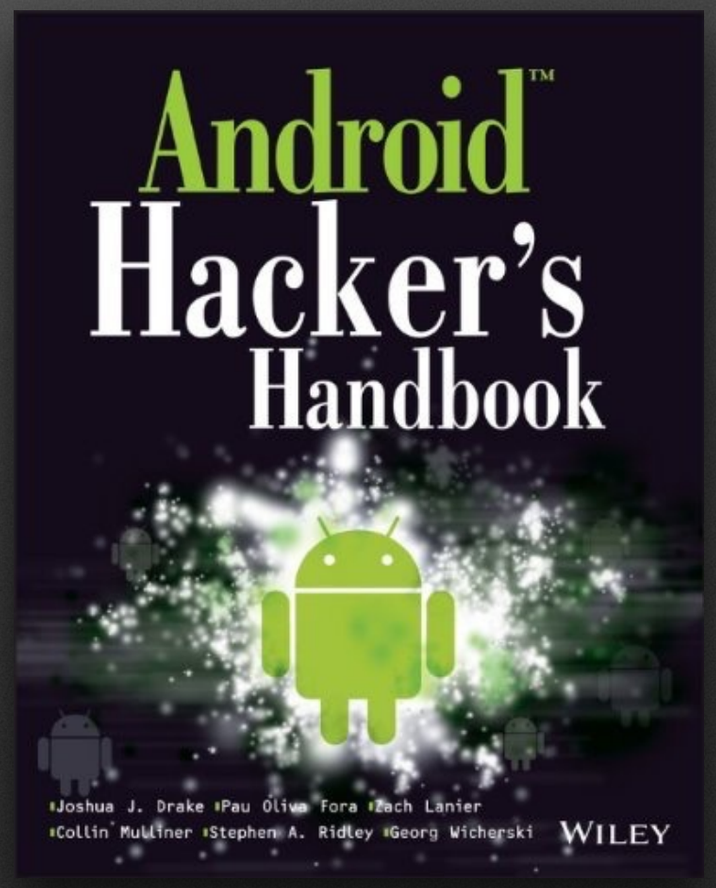
Assessing Android Applications Using Command-Line Fu

Pau Oliva Fora
@pof

$ whoami

- Pau Oliva Fora, aka @pof
  - Mobile Security Engineer with NowSecure
  - Linux guy, R+D background
  - Smartphone research since 2004
  - Android research since 2008
  - Co-Author of Android Hacker's Handbook



AGENDA

- Working with APK files:
  - Checking the app certificate
  - Getting permissions, manifest, resources, etc...
  - Disassembling the application
  - Decompiling the application
  - Obfuscation check
  - MasterKey exploit check
  - Checking for the SecureRandom bug
  - Other useful tips
- Interacting with installed APPs:
  - Obtaining application data
  - Checking for debuggable processes
  - Checking for debuggable apps

Working with APK files

App Certificate

- Multiple tools to check certificates:
  - OpenSSL

```
openssl pkcs7 -inform DER -in META-INF/*.RSA
-noout -print_certs -text
```

  - keytool

```
keytool -printcert -file META-INF/*.RSA
```

  - jarsigner

```
jarsigner -verify -certs -verbose *.apk
```

App Certificate

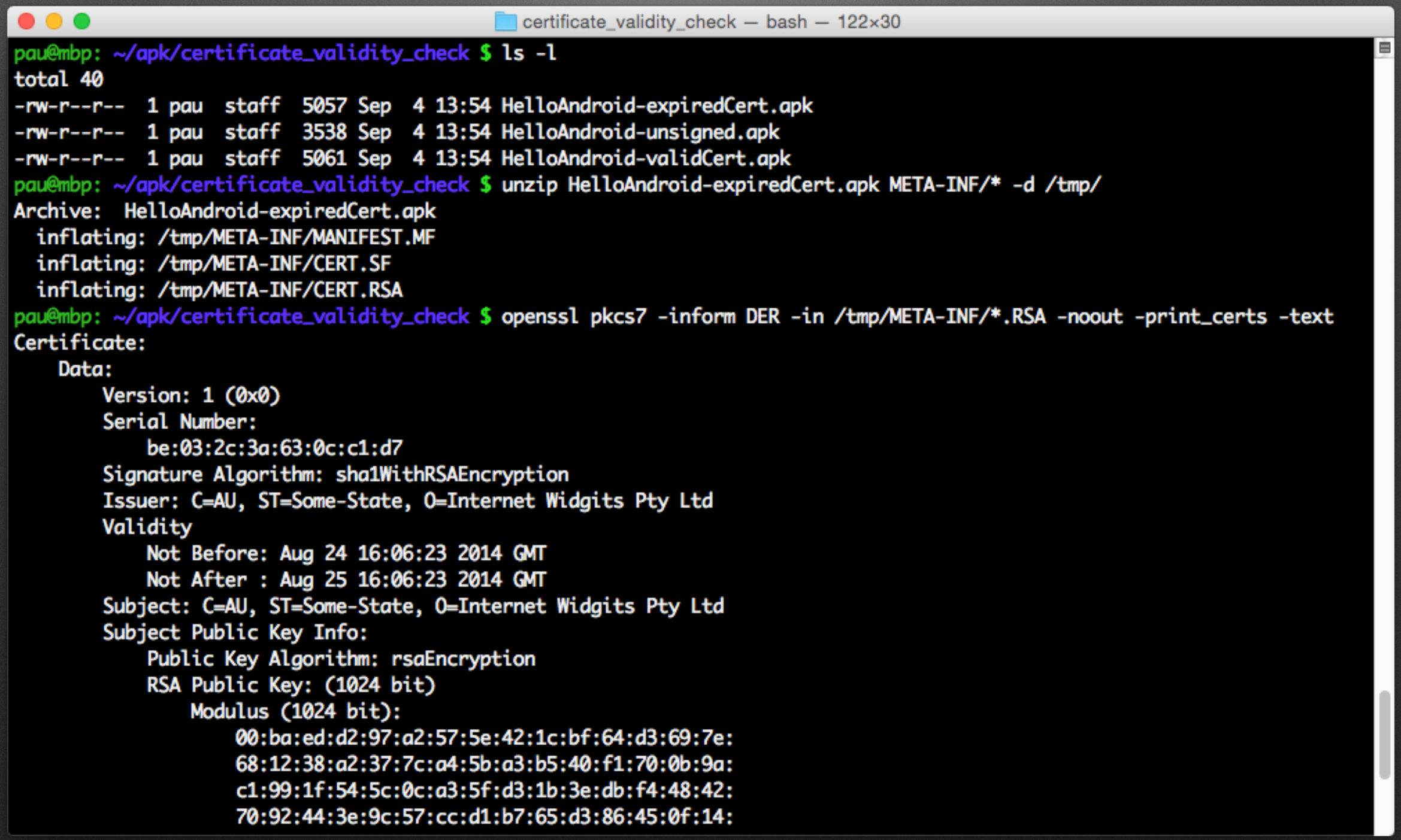
App Certificate

- Android M performs stricter validation of APKs:
  - An APK is considered corrupt if a file is declared in the MANIFEST.MF but not present in the APK itself
  - An APK must be re-signed if any of the contents are removed

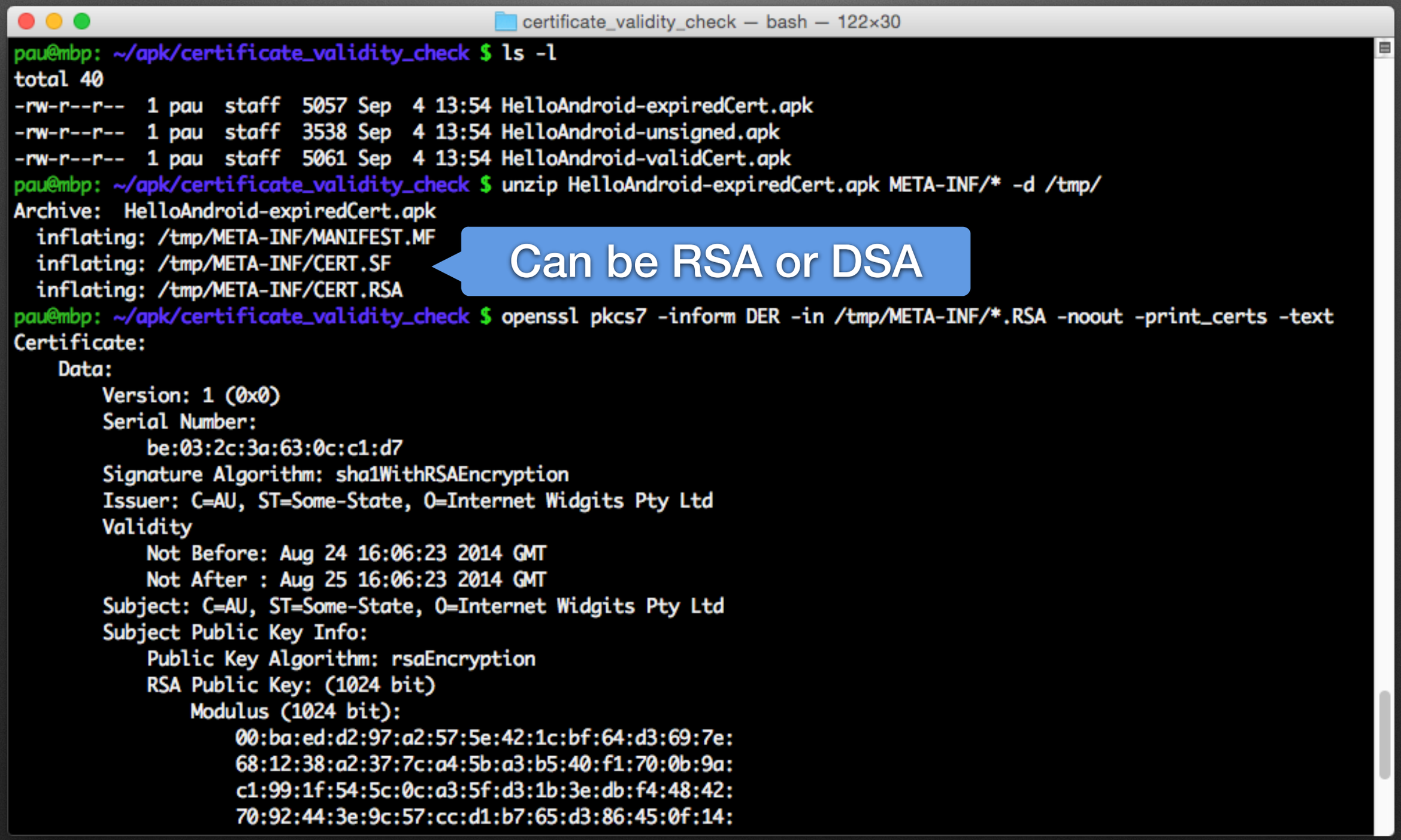
App Certificate

- Android M performs stricter validation of APKs:
  - An APK is considered corrupt if a file is declared in the MANIFEST.MF but not present in the APK itself
  - An APK must be re-signed if any of the contents are removed
- A common trick is to abuse the META-INF folder to stuff information into the APK without breaking the signature validation

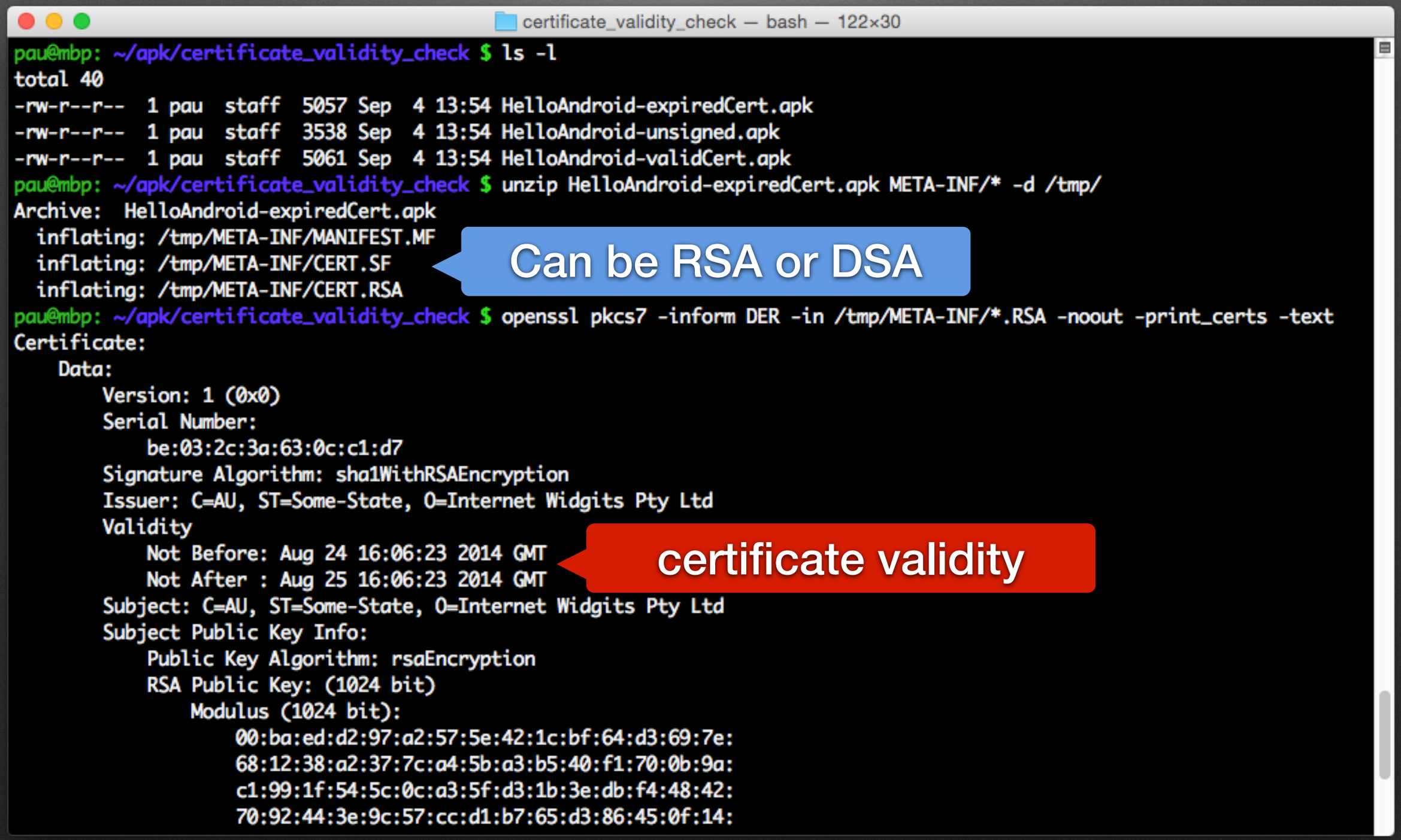
openssl


certificate_validity_check — bash — 122×30
pau@mbp: ~/apk/certificate_validity_check $ ls -l
total 40
-rw-r--r-- 1 pau staff 5057 Sep 4 13:54 HelloAndroid-expiredCert.apk
-rw-r--r-- 1 pau staff 3538 Sep 4 13:54 HelloAndroid-unsigned.apk
-rw-r--r-- 1 pau staff 5061 Sep 4 13:54 HelloAndroid-validCert.apk
pau@mbp: ~/apk/certificate_validity_check $ unzip HelloAndroid-expiredCert.apk META-INF/* -d /tmp/
Archive: HelloAndroid-expiredCert.apk
  inflating: /tmp/META-INF/MANIFEST.MF
  inflating: /tmp/META-INF/CERT.SF
  inflating: /tmp/META-INF/CERT.RSA
pau@mbp: ~/apk/certificate_validity_check $ openssl pkcs7 -inform DER -in /tmp/META-INF/*.RSA -noout -print_certs -text
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            be:03:2c:3a:63:0c:c1:d7
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Validity
            Not Before: Aug 24 16:06:23 2014 GMT
            Not After : Aug 25 16:06:23 2014 GMT
        Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:ba:ed:d2:97:a2:57:5e:42:1c:bf:64:d3:69:7e:
                    68:12:38:a2:37:7c:a4:5b:a3:b5:40:f1:70:0b:9a:
                    c1:99:1f:54:5c:0c:a3:5f:d3:1b:3e:db:f4:48:42:
                    70:92:44:3e:9c:57:cc:d1:b7:65:d3:86:45:0f:14:

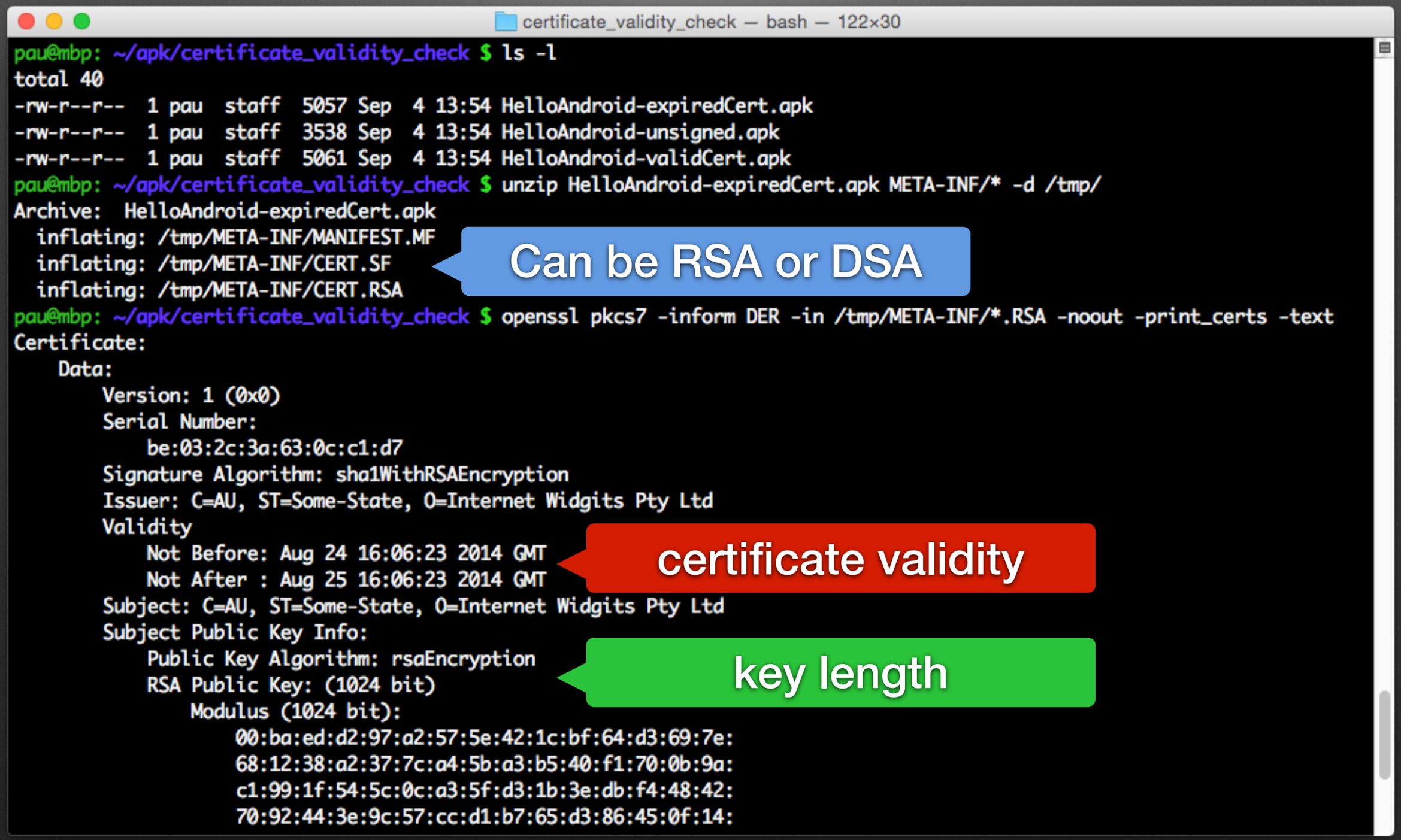
openssl


certificate_validity_check — bash — 122×30
pau@mbp: ~/apk/certificate_validity_check $ ls -l
total 40
-rw-r--r-- 1 pau staff 5057 Sep 4 13:54 HelloAndroid-expiredCert.apk
-rw-r--r-- 1 pau staff 3538 Sep 4 13:54 HelloAndroid-unsigned.apk
-rw-r--r-- 1 pau staff 5061 Sep 4 13:54 HelloAndroid-validCert.apk
pau@mbp: ~/apk/certificate_validity_check $ unzip HelloAndroid-expiredCert.apk META-INF/* -d /tmp/
Archive: HelloAndroid-expiredCert.apk
  inflating: /tmp/META-INF/MANIFEST.MF
  inflating: /tmp/META-INF/CERT.SF
  inflating: /tmp/META-INF/CERT.RSA
Can be RSA or DSA
pau@mbp: ~/apk/certificate_validity_check $ openssl pkcs7 -inform DER -in /tmp/META-INF/*.RSA -noout -print_certs -text
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            be:03:2c:3a:63:0c:c1:d7
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Validity
            Not Before: Aug 24 16:06:23 2014 GMT
            Not After : Aug 25 16:06:23 2014 GMT
        Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:ba:ed:d2:97:a2:57:5e:42:1c:bf:64:d3:69:7e:
                    68:12:38:a2:37:7c:a4:5b:a3:b5:40:f1:70:0b:9a:
                    c1:99:1f:54:5c:0c:a3:5f:d3:1b:3e:db:f4:48:42:
                    70:92:44:3e:9c:57:cc:d1:b7:65:d3:86:45:0f:14:

openssl


certificate_validity_check — bash — 122×30
pau@mbp: ~/apk/certificate_validity_check $ ls -l
total 40
-rw-r--r-- 1 pau staff 5057 Sep 4 13:54 HelloAndroid-expiredCert.apk
-rw-r--r-- 1 pau staff 3538 Sep 4 13:54 HelloAndroid-unsigned.apk
-rw-r--r-- 1 pau staff 5061 Sep 4 13:54 HelloAndroid-validCert.apk
pau@mbp: ~/apk/certificate_validity_check $ unzip HelloAndroid-expiredCert.apk META-INF/* -d /tmp/
Archive: HelloAndroid-expiredCert.apk
  inflating: /tmp/META-INF/MANIFEST.MF
  inflating: /tmp/META-INF/CERT.SF
  inflating: /tmp/META-INF/CERT.RSA
Can be RSA or DSA
pau@mbp: ~/apk/certificate_validity_check $ openssl pkcs7 -inform DER -in /tmp/META-INF/*.RSA -noout -print_certs -text
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            be:03:2c:3a:63:0c:c1:d7
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Validity
            Not Before: Aug 24 16:06:23 2014 GMT
            Not After : Aug 25 16:06:23 2014 GMT
certificate validity
        Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:ba:ed:d2:97:a2:57:5e:42:1c:bf:64:d3:69:7e:
                    68:12:38:a2:37:7c:a4:5b:a3:b5:40:f1:70:0b:9a:
                    c1:99:1f:54:5c:0c:a3:5f:d3:1b:3e:db:f4:48:42:
                    70:92:44:3e:9c:57:cc:d1:b7:65:d3:86:45:0f:14:

openssl


certificate_validity_check — bash — 122×30
pau@mbp: ~/apk/certificate_validity_check $ ls -l
total 40
-rw-r--r-- 1 pau staff 5057 Sep 4 13:54 HelloAndroid-expiredCert.apk
-rw-r--r-- 1 pau staff 3538 Sep 4 13:54 HelloAndroid-unsigned.apk
-rw-r--r-- 1 pau staff 5061 Sep 4 13:54 HelloAndroid-validCert.apk
pau@mbp: ~/apk/certificate_validity_check $ unzip HelloAndroid-expiredCert.apk META-INF/* -d /tmp/
Archive: HelloAndroid-expiredCert.apk
  inflating: /tmp/META-INF/MANIFEST.MF
  inflating: /tmp/META-INF/CERT.SF
  inflating: /tmp/META-INF/CERT.RSA
Can be RSA or DSA
pau@mbp: ~/apk/certificate_validity_check $ openssl pkcs7 -inform DER -in /tmp/META-INF/*.RSA -noout -print_certs -text
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            be:03:2c:3a:63:0c:c1:d7
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Validity
            Not Before: Aug 24 16:06:23 2014 GMT
            Not After : Aug 25 16:06:23 2014 GMT
certificate validity
        Subject: C=AU, ST=Some-State, O=Internet Widgits Pty Ltd
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
key length
                Modulus (1024 bit):
                    00:ba:ed:d2:97:a2:57:5e:42:1c:bf:64:d3:69:7e:
                    68:12:38:a2:37:7c:a4:5b:a3:b5:40:f1:70:0b:9a:
                    c1:99:1f:54:5c:0c:a3:5f:d3:1b:3e:db:f4:48:42:
                    70:92:44:3e:9c:57:cc:d1:b7:65:d3:86:45:0f:14:

keytool

apk — bash — 122×30

```
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ unzip HelloAndroid.apk META-INF/*
Archive:  HelloAndroid.apk
  inflating: META-INF/MANIFEST.MF
  inflating: META-INF/CERT.SF
  inflating: META-INF/CERT.RSA
pau@mbp: /tmp/apk $ keytool -printcert -file META-INF/*.RSA
Owner: O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
Issuer: O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
Serial number: a129190637477722
Valid from: Tue Aug 26 18:04:14 CEST 2014 until: Fri Jan 10 17:04:14 CET 2042
Certificate fingerprints:
         MD5:  3D:7A:CA:0D:66:CF:51:A0:3D:29:0A:8B:EA:FC:3F:D3
         SHA1: B7:EE:29:C6:B7:FF:93:B3:9C:CF:E9:96:06:EB:EF:45:1A:57:09:90
         SHA256: C0:DB:B6:CB:4D:8E:05:C7:50:85:74:8A:54:28:01:7B:98:4C:BD:BA:18:C3:58:72:9D:5E:D3:7B:CB:B5:0A:79
         Signature algorithm name: SHA1withRSA
         Version: 1
pau@mbp: /tmp/apk $
```

jarsigner

apk — bash — 122×30

```
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ jarsigner -verify -certs -verbose HelloAndroid.apk

s        335 Tue Aug 26 18:04:14 CEST 2014 META-INF/MANIFEST.MF

      X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
      [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]

        388 Tue Aug 26 18:04:14 CEST 2014 META-INF/CERT.SF
        823 Tue Aug 26 18:04:14 CEST 2014 META-INF/CERT.RSA
sm      696 Tue Aug 26 18:04:14 CEST 2014 res/layout/main.xml

      X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
      [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]

sm     1256 Tue Aug 26 18:04:14 CEST 2014 AndroidManifest.xml

      X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
      [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]

sm      764 Tue Aug 26 17:43:08 CEST 2014 resources.arsc

      X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
      [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
      [CertPath not validated: Path does not chain with any of the trust anchors]
```

jarsigner

apk — bash — 122×30

```
     X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
     [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
     [CertPath not validated: Path does not chain with any of the trust anchors]

sm       764 Tue Aug 26 17:43:08 CEST 2014 resources.arsc

     X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
     [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
     [CertPath not validated: Path does not chain with any of the trust anchors]

sm      3376 Tue Aug 26 18:04:14 CEST 2014 classes.dex

     X.509, O=Internet Widgits Pty Ltd, ST=Some-State, C=AU
     [certificate is valid from 8/26/14 6:04 PM to 1/10/42 5:04 PM]
     [CertPath not validated: Path does not chain with any of the trust anchors]


  s = signature was verified
  m = entry is listed in manifest
  k = at least one certificate was found in keystore
  i = at least one certificate was found in identity scope

jar verified.

Warning:
This jar contains entries whose certificate chain is not validated.
This jar contains signatures that does not include a timestamp. Without a timestamp, users may not be able to validate thi
s jar after the signer certificate's expiration date (2042-01-10) or after any future revocation date.
pau@mbp: /tmp/apk $
```

Android Asset Packaging Tool: aapt

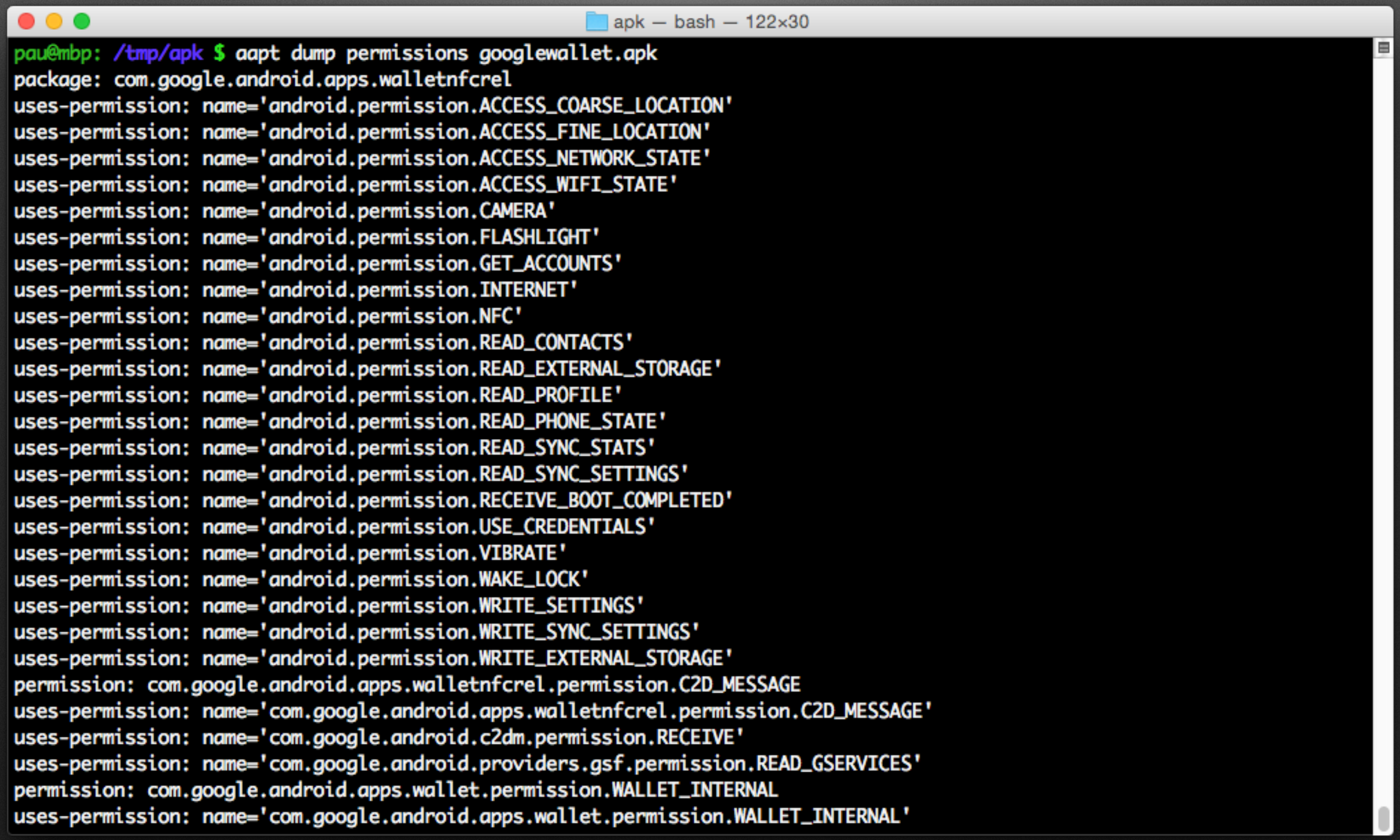
- From Android SDK build-tools
- Command-line tool to work with APKs

Android Asset Packaging Tool: aapt

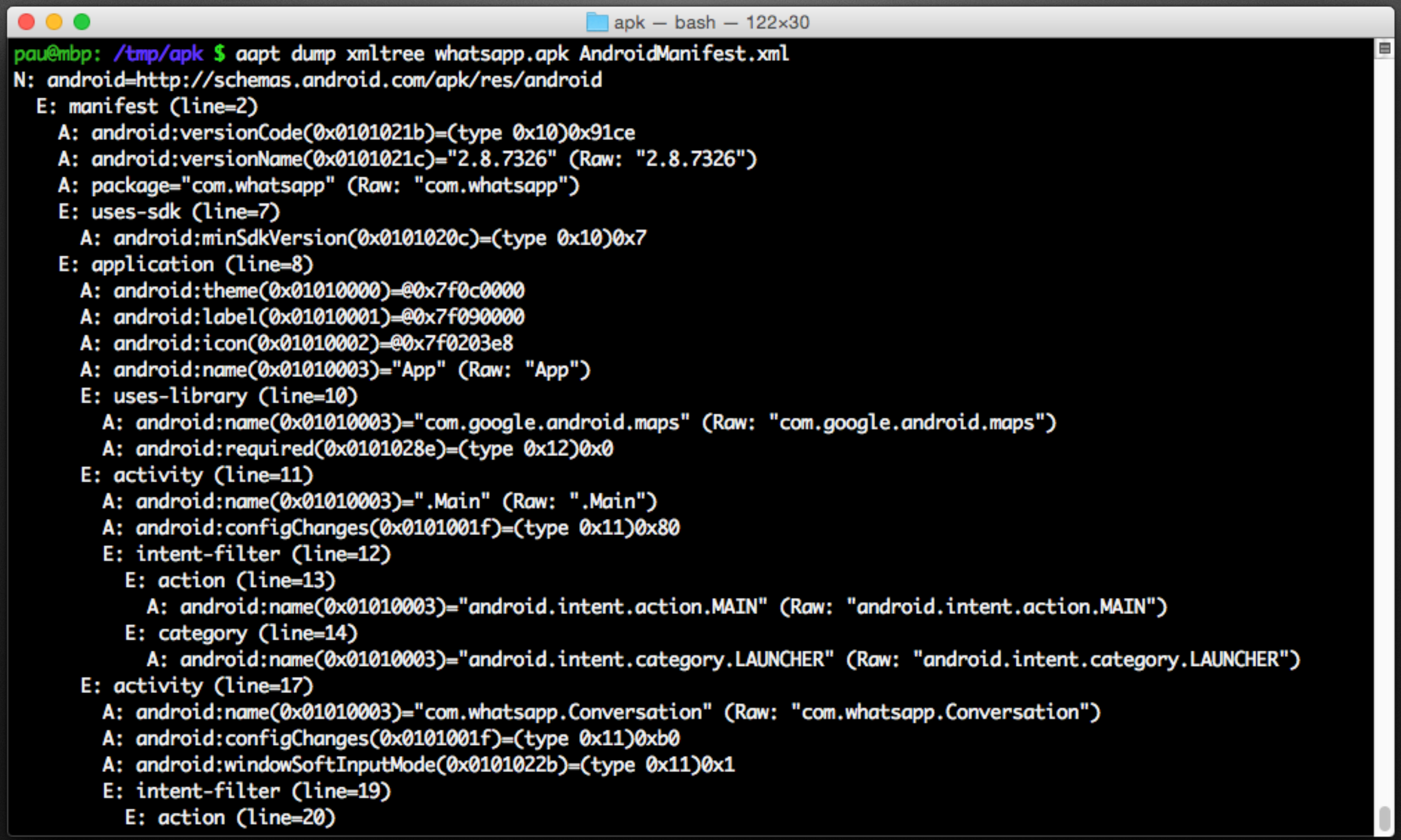
pau — bash — 122×30

```
pau@mbp: ~ $ aapt 2>&1 |head -n 15
Android Asset Packaging Tool

Usage:
 aapt l[ist] [-v] [-a] file.{zip,jar,apk}
   List contents of Zip-compatible archive.

 aapt d[ump] [--values] [--include-meta-data] WHAT file.{apk} [asset [asset ...]]
   strings           Print the contents of the resource table string pool in the APK.
   badging           Print the label and icon for the app declared in APK.
   permissions       Print the permissions from the APK.
   resources         Print the resource table from the APK.
   configurations    Print the configurations in the APK.
   xmltree           Print the compiled xmls in the given assets.
   xmlstrings        Print the strings of the given compiled xml assets.

pau@mbp: ~ $
```

Android Asset Packaging Tool: aapt



Android Asset Packaging Tool: aapt

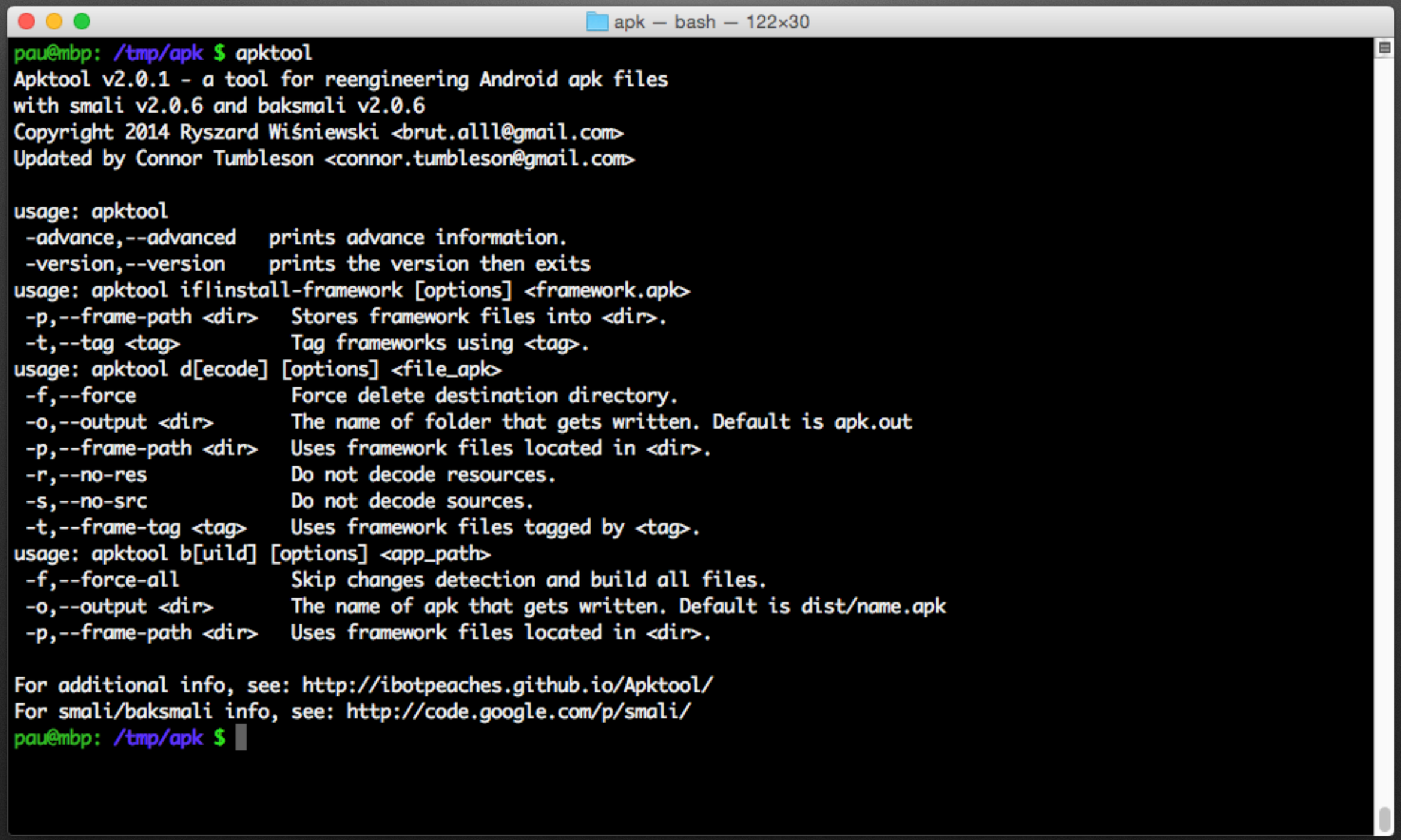


apktool

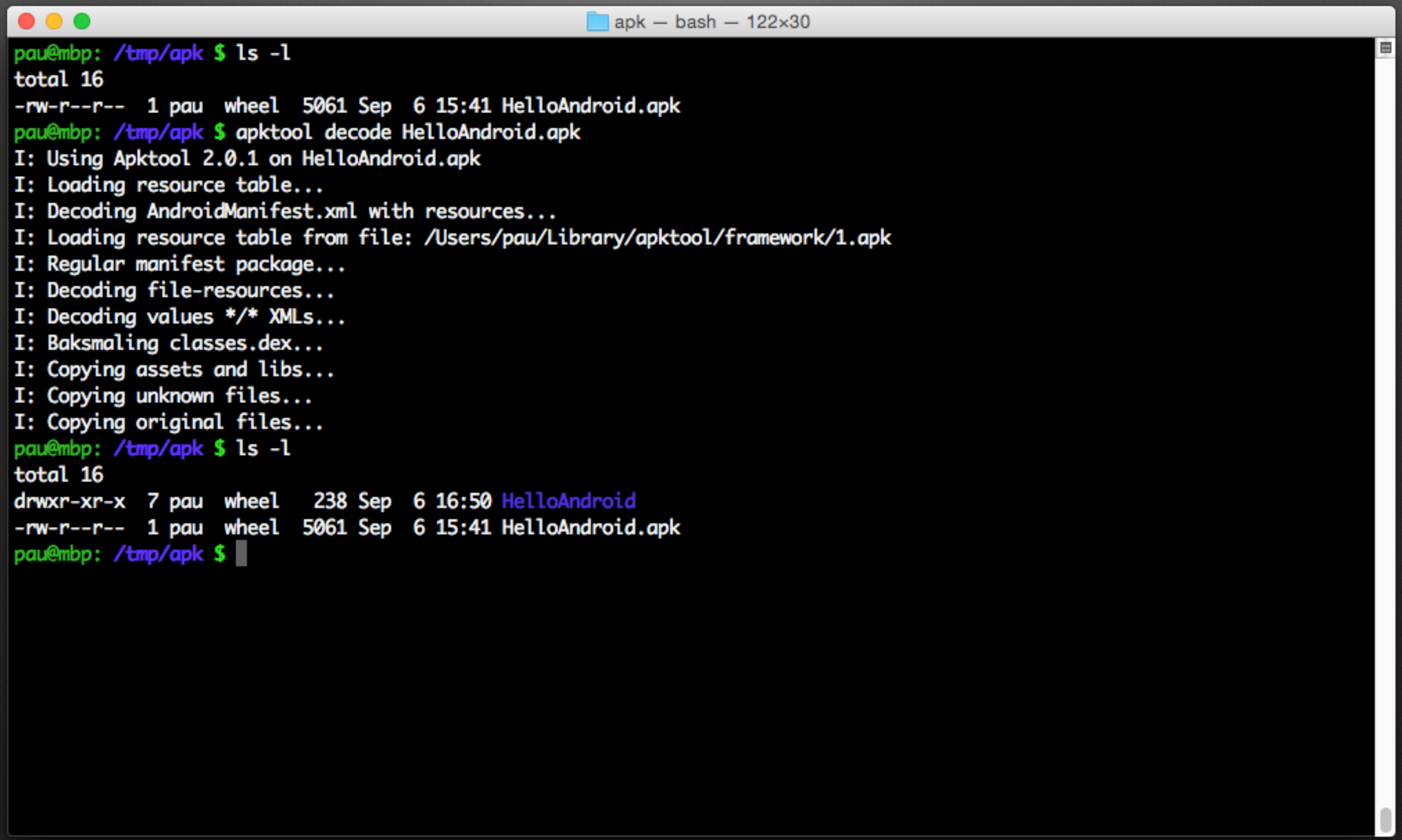
- A tool for reverse engineering Android APK files
  - Decode resources to original form
  - Get smali source
  - Rebuild APK back to binary after modifications

http://ibotpeaches.github.io/Apktool/

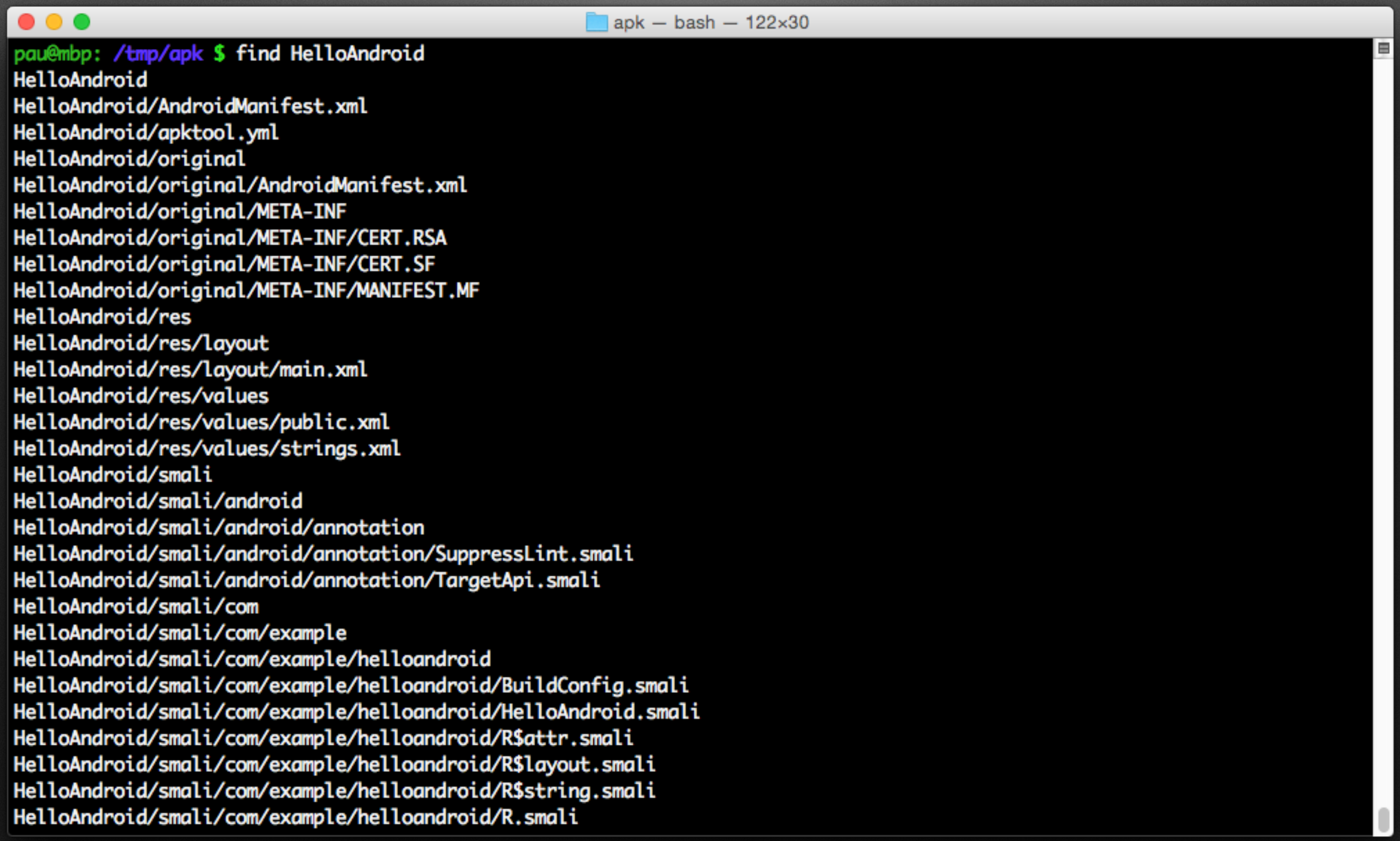
apktool


apk — bash — 122×30
pau@mbp: /tmp/apk $ apktool
Apktool v2.0.1 - a tool for reengineering Android apk files
with smali v2.0.6 and baksmali v2.0.6
Copyright 2014 Ryszard Wiśniewski <brut.alll@gmail.com>
Updated by Connor Tumbleson <connor.tumbleson@gmail.com>
usage: apktool
-advance,--advanced prints advance information.
-version,--version prints the version then exits
usage: apktool if|install-framework [options] <framework.apk>
-p,--frame-path <dir> Stores framework files into <dir>.
-t,--tag <tag> Tag frameworks using <tag>.
usage: apktool d[ecode] [options] <file_apk>
-f,--force Force delete destination directory.
-o,--output <dir> The name of folder that gets written. Default is apk.out
-p,--frame-path <dir> Uses framework files located in <dir>.
-r,--no-res Do not decode resources.
-s,--no-src Do not decode sources.
-t,--frame-tag <tag> Uses framework files tagged by <tag>.
usage: apktool b[uild] [options] <app_path>
-f,--force-all Skip changes detection and build all files.
-o,--output <dir> The name of apk that gets written. Default is dist/name.apk
-p,--frame-path <dir> Uses framework files located in <dir>.
For additional info, see: http://ibotpeaches.github.io/Apktool/
For smali/baksmali info, see: http://code.google.com/p/smali/
pau@mbp: /tmp/apk $

apktool


apk — bash — 122×30
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ apktool decode HelloAndroid.apk
I: Using Apktool 2.0.1 on HelloAndroid.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /Users/pau/Library/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
pau@mbp: /tmp/apk $ ls -l
total 16
drwxr-xr-x 7 pau wheel   238 Sep 6 16:50 HelloAndroid
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $

apktool


apk — bash — 122×30
pau@mbp: /tmp/apk $ find HelloAndroid
HelloAndroid
HelloAndroid/AndroidManifest.xml
HelloAndroid/apktool.yml
HelloAndroid/original
HelloAndroid/original/AndroidManifest.xml
HelloAndroid/original/META-INF
HelloAndroid/original/META-INF/CERT.RSA
HelloAndroid/original/META-INF/CERT.SF
HelloAndroid/original/META-INF/MANIFEST.MF
HelloAndroid/res
HelloAndroid/res/layout
HelloAndroid/res/layout/main.xml
HelloAndroid/res/values
HelloAndroid/res/values/public.xml
HelloAndroid/res/values/strings.xml
HelloAndroid/smali
HelloAndroid/smali/android
HelloAndroid/smali/android/annotation
HelloAndroid/smali/android/annotation/SuppressLint.smali
HelloAndroid/smali/android/annotation/TargetApi.smali
HelloAndroid/smali/com
HelloAndroid/smali/com/example
HelloAndroid/smali/com/example/helloandroid
HelloAndroid/smali/com/example/helloandroid/BuildConfig.smali
HelloAndroid/smali/com/example/helloandroid/HelloAndroid.smali
HelloAndroid/smali/com/example/helloandroid/R$attr.smali
HelloAndroid/smali/com/example/helloandroid/R$layout.smali
HelloAndroid/smali/com/example/helloandroid/R$string.smali
HelloAndroid/smali/com/example/helloandroid/R.smali

apktool

apk — vim — 122×30

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.example.helloandroid">
    <application android:label="@string/app_name">
        <activity android:label="@string/app_name" android:name="HelloAndroid">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"HelloAndroid/AndroidManifest.xml" 11L, 543C
```

apktool

apk — vim — 122×30

```
.class public Lcom/example/helloandroid/HelloAndroid;
.super Landroid/app/Activity;
.source "HelloAndroid.java"


# direct methods
.method public constructor <init>()V
    .locals 0

    .prologue
    .line 9
    invoke-direct {p0}, Landroid/app/Activity;-><init>()V

    return-void
.end method


# virtual methods
.method public onCreate(Landroid/os/Bundle;)V
    .locals 2
    .param p1, "savedInstanceState"    # Landroid/os/Bundle;

    .prologue
    .line 15
    invoke-super {p0, p1}, Landroid/app/Activity;->onCreate(Landroid/os/Bundle;)V

    .line 16
    new-instance v0, Landroid/widget/TextView;

"HelloAndroid/smali/com/example/helloandroid/HelloAndroid.smali" 58L, 1408C
```

Decompiling

- A bunch of decompilers available, each producing different results on some situations
- What usually "works best" for me:
  - JEB (commercial) - https://www.pnfsoftware.com/
  - jadx (DEX → JAVA) - https://github.com/skylot/jadx
  - enjarify + jad (DEX → JAR → JAVA)
    - https://github.com/google/enjarify
    - http://varaneckas.com/jad/

Decompiling

- A bunch of decompilers available, each producing different results on some situations

- What usually "works best" for me:

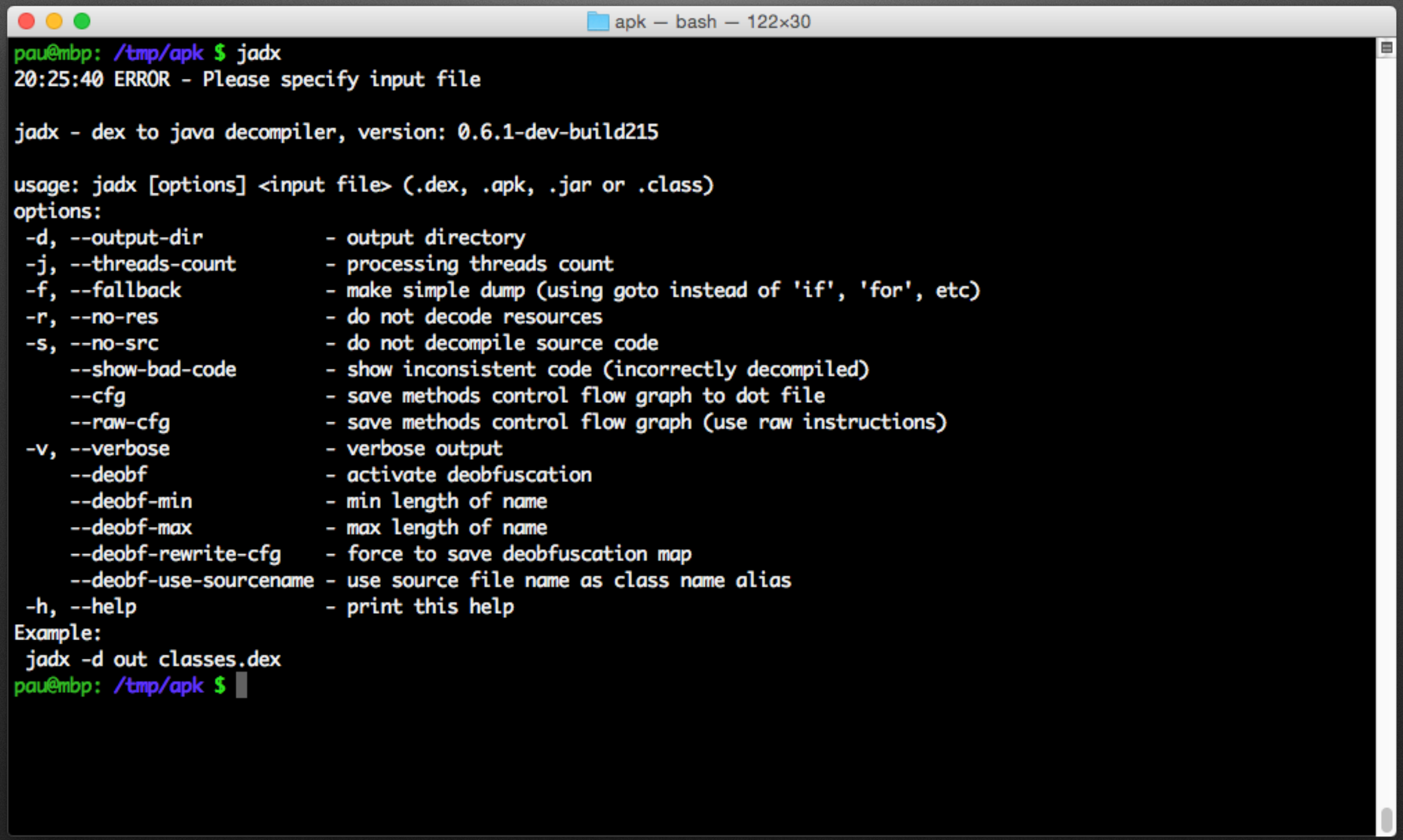
  - JEB (commercial) - https://www.pnfsoftware.com/

  - jadx (DEX → JAVA) - https://github.com/skylot/jadx

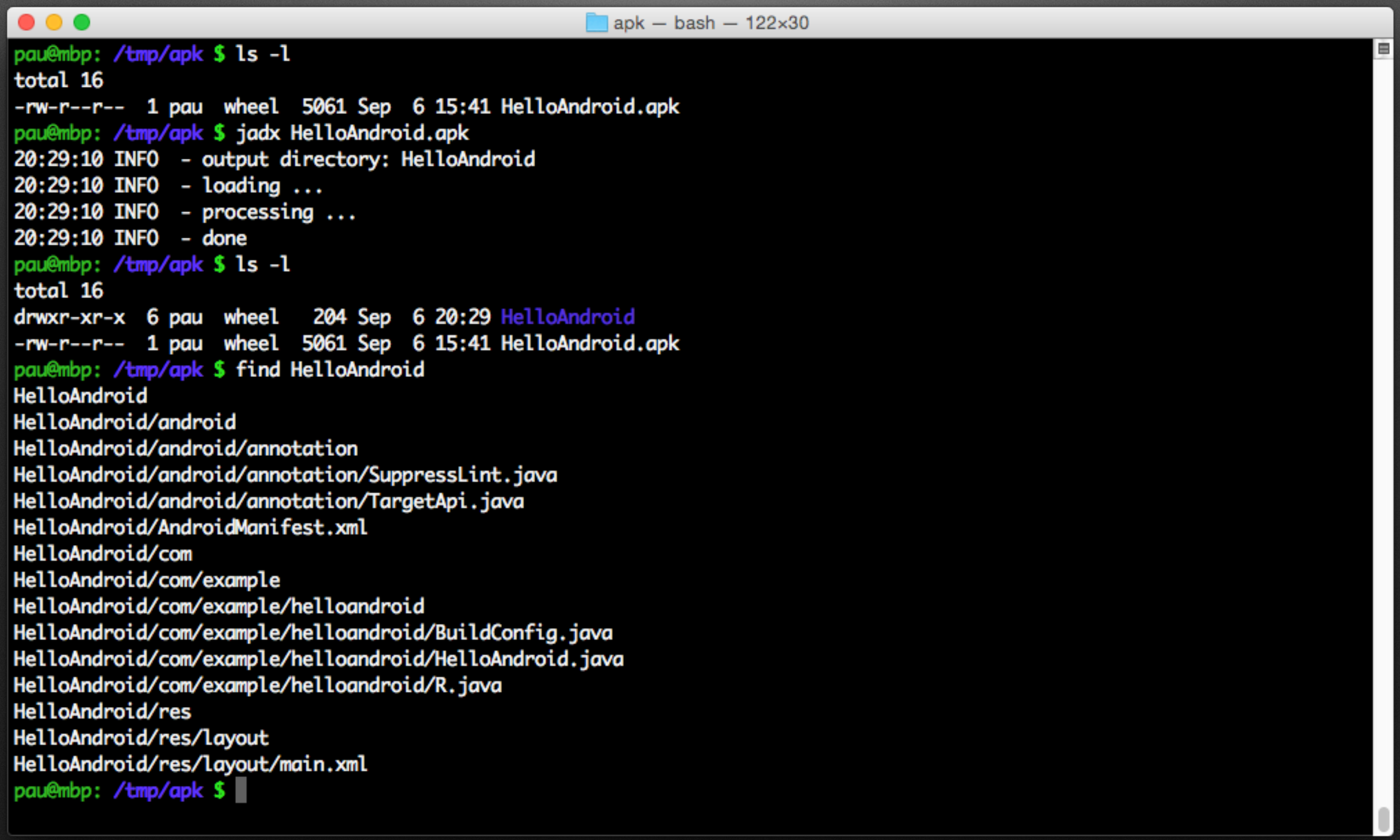
  - enjarify + replaces the old dex2jar
    - https://github.com/google/enjarify

    - http://varaneckas.com/jad/

jadx


apk — bash — 122×30
pau@mbp: /tmp/apk $ jadx
20:25:40 ERROR - Please specify input file
jadx - dex to java decompiler, version: 0.6.1-dev-build215
usage: jadx [options] <input file> (.dex, .apk, .jar or .class)
options:
-d, --output-dir - output directory
-j, --threads-count - processing threads count
-f, --fallback - make simple dump (using goto instead of 'if', 'for', etc)
-r, --no-res - do not decode resources
-s, --no-src - do not decompile source code
--show-bad-code - show inconsistent code (incorrectly decompiled)
--cfg - save methods control flow graph to dot file
--raw-cfg - save methods control flow graph (use raw instructions)
-v, --verbose - verbose output
--deobf - activate deobfuscation
--deobf-min - min length of name
--deobf-max - max length of name
--deobf-rewrite-cfg - force to save deobfuscation map
--deobf-use-sourcename - use source file name as class name alias
-h, --help - print this help
Example:
jadx -d out classes.dex
pau@mbp: /tmp/apk $

jadx


apk — bash — 122×30
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ jadx HelloAndroid.apk
20:29:10 INFO - output directory: HelloAndroid
20:29:10 INFO - loading ...
20:29:10 INFO - processing ...
20:29:10 INFO - done
pau@mbp: /tmp/apk $ ls -l
total 16
drwxr-xr-x 6 pau wheel   204 Sep 6 20:29 HelloAndroid
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ find HelloAndroid
HelloAndroid
HelloAndroid/android
HelloAndroid/android/annotation
HelloAndroid/android/annotation/SuppressLint.java
HelloAndroid/android/annotation/TargetApi.java
HelloAndroid/AndroidManifest.xml
HelloAndroid/com
HelloAndroid/com/example
HelloAndroid/com/example/helloandroid
HelloAndroid/com/example/helloandroid/BuildConfig.java
HelloAndroid/com/example/helloandroid/HelloAndroid.java
HelloAndroid/com/example/helloandroid/R.java
HelloAndroid/res
HelloAndroid/res/layout
HelloAndroid/res/layout/main.xml
pau@mbp: /tmp/apk $

jadx

apk — bash — 122×30

```
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ jadx HelloAndroid.apk
20:29:10 INFO  - output directory: HelloAndroid
20:29:10 INFO  - loading ...
20:29:10 INFO  - processing ...
20:29:10 INFO  - done
pau@mbp: /tmp/apk $ ls -l
total 16
drwxr-xr-x  6 pau  wheel   204 Sep  6 20:29 HelloAndroid
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ find HelloAndroid
HelloAndroid
HelloAndroid/android
HelloAndroid/android/annotation
HelloAndroid/android/annotation/SuppressLint.java
HelloAndroid/android/annotation/TargetApi.java
HelloAndroid/AndroidManifest.xml
HelloAndroid/com
HelloAndroid/com/example
HelloAndroid/com/example/helloandroid
HelloAndroid/com/example/helloandroid/BuildConfig.java
HelloAndroid/com/example/helloandroid/HelloAndroid.java
HelloAndroid/com/example/helloandroid/R.java
HelloAndroid/res
HelloAndroid/res/layout
HelloAndroid/res/layout/main.xml
pau@mbp: /tmp/apk $
```

decompiled source

jadx

apk — bash — 122×30

```
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ jadx HelloAndroid.apk
20:29:10 INFO  - output directory: HelloAndroid
20:29:10 INFO  - loading ...
20:29:10 INFO  - processing ...
20:29:10 INFO  - done
pau@mbp: /tmp/apk $ ls -l
total 16
drwxr-xr-x  6 pau  wheel   204 Sep  6 20:29 HelloAndroid
-rw-r--r--  1 pau  wheel  5061 Sep  6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ find HelloAndroid
HelloAndroid
HelloAndroid/android
HelloAndroid/android/annotation
HelloAndroid/android/annotation/SuppressLint.java
HelloAndroid/android/annotation/TargetApi.java
HelloAndroid/AndroidManifest.xml
HelloAndroid/com
HelloAndroid/com/example
HelloAndroid/com/example/helloandroid
HelloAndroid/com/example/helloandroid/BuildConfig.java
HelloAndroid/com/example/helloandroid/HelloAndroid.java
HelloAndroid/com/example/helloandroid/R.java
HelloAndroid/res
HelloAndroid/res/layout
HelloAndroid/res/layout/main.xml
pau@mbp: /tmp/apk $
```

decoded resources

decompiled source

jadx

apk — vim — 122×30

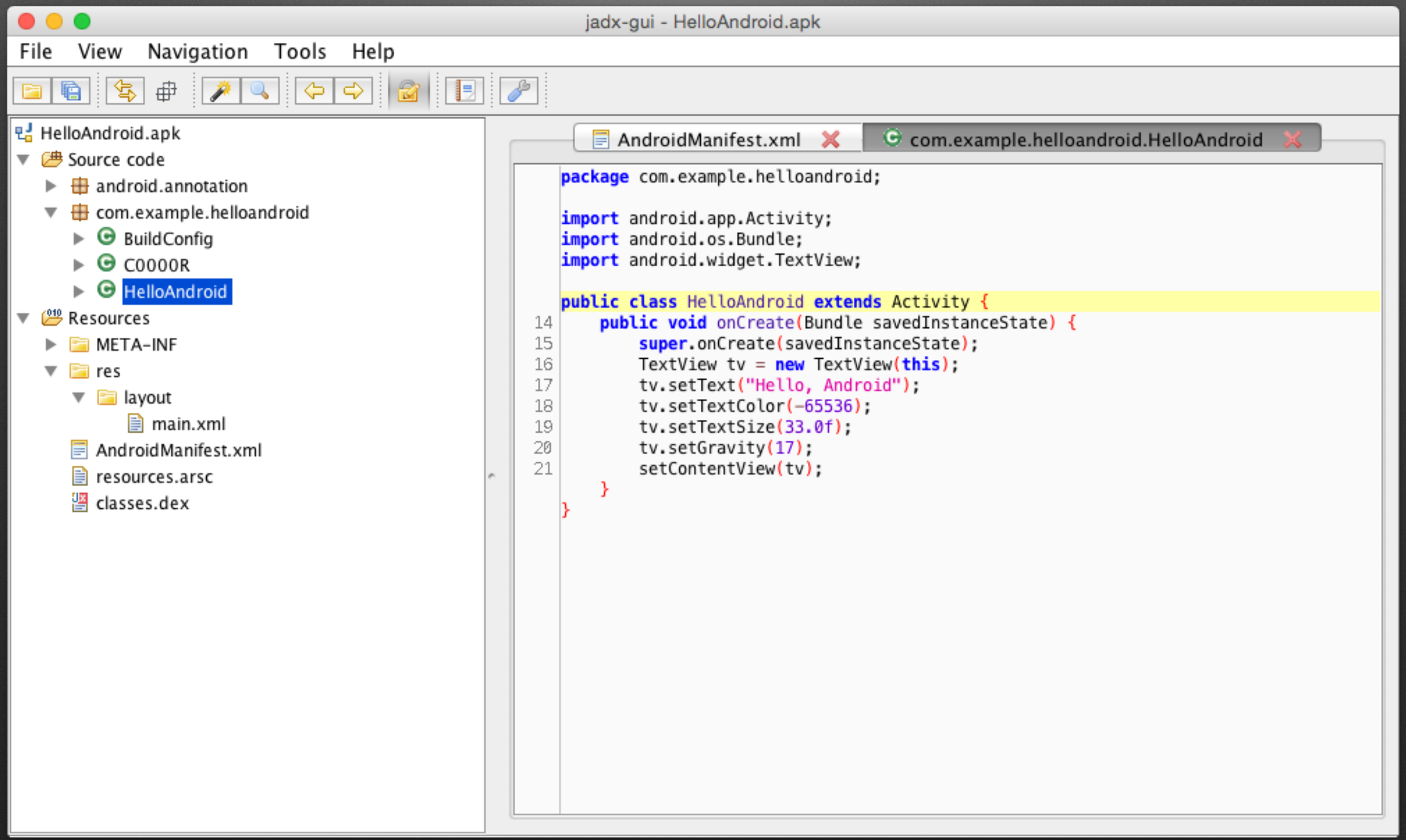
```
package com.example.helloandroid;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloAndroid extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(this);
        tv.setText("Hello, Android");
        tv.setTextColor(-65536);
        tv.setTextSize(33.0f);
        tv.setGravity(17);
        setContentView(tv);
    }
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

HelloAndroid/com/example/helloandroid/HelloAndroid.java

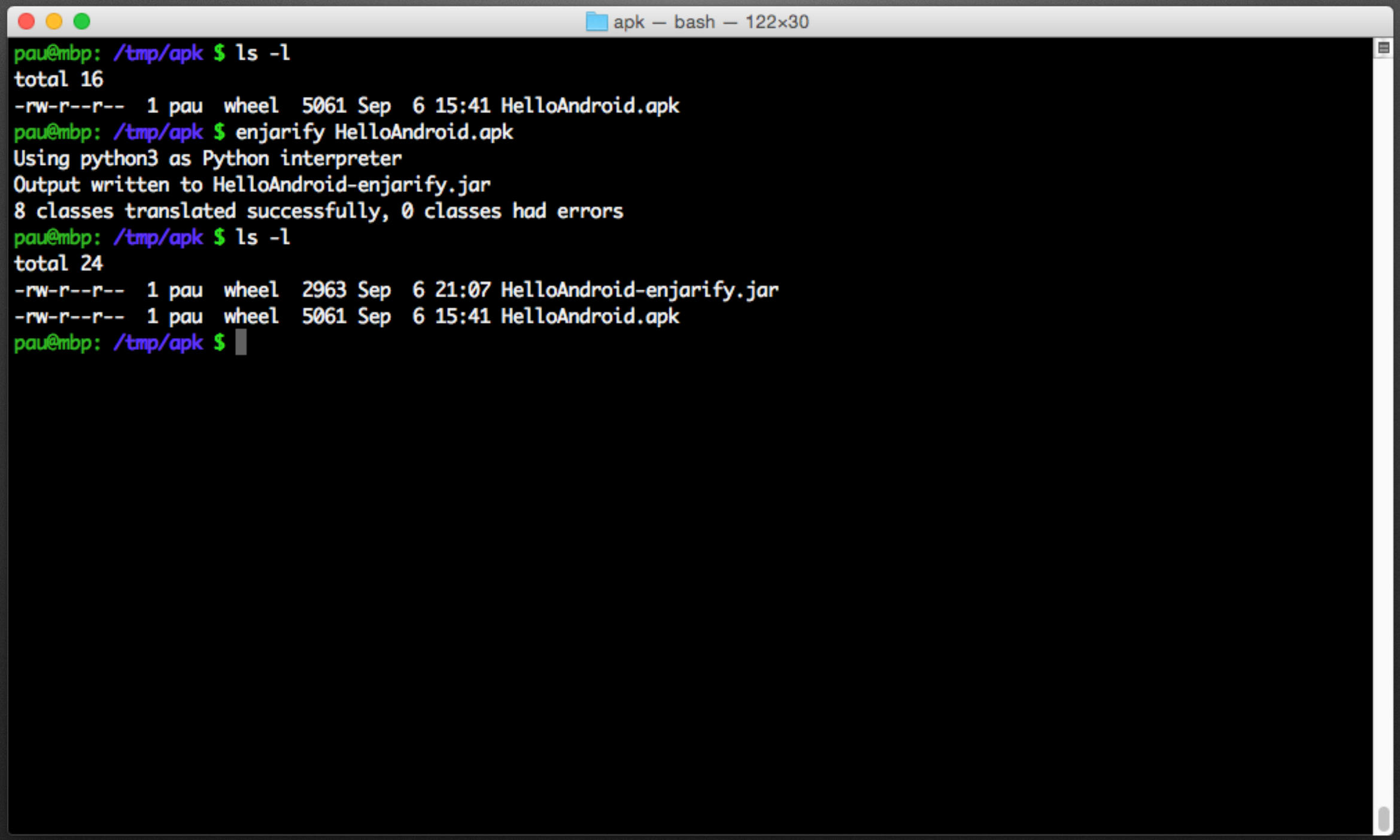
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:"http://schemas.android.com/apk/res/android"
 android:versionCode="1" android:versionName="1.0" package="
com.example.helloandroid">
    <application android:label="@string/app_name">
        <activity android:label="@string/app_name" android:n
ame="HelloAndroid">
            <intent-filter>
                <action android:name="android.intent.action.
MAIN" />
                <category android:name="android.intent.categ
ory.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

HelloAndroid/AndroidManifest.xml

jadx-gui


jadx-gui - HelloAndroid.apk
File View Navigation Tools Help
HelloAndroid.apk
Source code
android.annotation
com.example.helloandroid
BuildConfig
C0000R
HelloAndroid
Resources
META-INF
res
layout
main.xml
AndroidManifest.xml
resources.arsc
classes.dex
AndroidManifest.xml
com.example.helloandroid.HelloAndroid
package com.example.helloandroid;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;
public class HelloAndroid extends Activity {
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
TextView tv = new TextView(this);
tv.setText("Hello, Android");
tv.setTextColor(-65536);
tv.setTextSize(33.0f);
tv.setGravity(17);
setContentView(tv);
}
}

enjarify


apk — bash — 122×30
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $ enjarify HelloAndroid.apk
Using python3 as Python interpreter
Output written to HelloAndroid-enjarify.jar
8 classes translated successfully, 0 classes had errors
pau@mbp: /tmp/apk $ ls -l
total 24
-rw-r--r-- 1 pau wheel 2963 Sep 6 21:07 HelloAndroid-enjarify.jar
-rw-r--r-- 1 pau wheel 5061 Sep 6 15:41 HelloAndroid.apk
pau@mbp: /tmp/apk $

jad

apk — bash — 122×30

```
pau@mbp: /tmp/apk $ ls -l
total 16
-rw-r--r--  1 pau  wheel  5061 Sep  9 23:06 HelloAndroid.apk
pau@mbp: /tmp/apk $ enjarify HelloAndroid.apk
Using python3 as Python interpreter
Output written to HelloAndroid-enjarify.jar
8 classes translated successfully, 0 classes had errors
pau@mbp: /tmp/apk $ ls -l
total 24
-rw-r--r--  1 pau  wheel  2963 Sep  9 23:08 HelloAndroid-enjarify.jar
-rw-r--r--  1 pau  wheel  5061 Sep  9 23:06 HelloAndroid.apk
pau@mbp: /tmp/apk $ unzip HelloAndroid-enjarify.jar
Archive:  HelloAndroid-enjarify.jar
 extracting: android/annotation/SuppressLint.class
 extracting: com/example/helloandroid/R$attr.class
 extracting: com/example/helloandroid/R$layout.class
 extracting: android/annotation/TargetApi.class
 extracting: com/example/helloandroid/R$string.class
 extracting: com/example/helloandroid/HelloAndroid.class
 extracting: com/example/helloandroid/R.class
 extracting: com/example/helloandroid/BuildConfig.class
pau@mbp: /tmp/apk $ jad -o -r -sjava -dsrc './**/*.class'
Parsing ./android/annotation/SuppressLint.class... Generating src/android/annotation/SuppressLint.java
Parsing ./android/annotation/TargetApi.class... Generating src/android/annotation/TargetApi.java
Parsing ./com/example/helloandroid/BuildConfig.class... Generating src/com/example/helloandroid/BuildConfig.java
Parsing ./com/example/helloandroid/HelloAndroid.class... Generating src/com/example/helloandroid/HelloAndroid.java
Parsing ./com/example/helloandroid/R.class... Generating src/com/example/helloandroid/R.java
pau@mbp: /tmp/apk $
```

jad

apk — vim — 122×30

```
// Decompiled by Jad v1.5.8g. Copyright 2001 Pavel Kouznetsov.
// Jad home page: http://www.kpdus.com/jad.html
// Decompiler options: packimports(3)

package com.example.helloandroid;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloAndroid extends Activity
{

    public HelloAndroid()
    {
    }

    public void onCreate(Bundle bundle)
    {
        super.onCreate(bundle);
        TextView textview = JVM INSTR new #14  <Class TextView>;
        textview.TextView(this);
        textview.setText("Hello, Android");
        textview.setTextColor(0xffff0000);
        textview.setTextSize(33F);
        textview.setGravity(17);
        setContentView(textview);
    }
}
"src/com/example/helloandroid/HelloAndroid.java" 29L, 732C
```

Obfuscation checks

- **ProGuard:**
  - file shrinker: detects and removes unused classes, fields, methods, and attributes
  - optimizer: optimizes bytecode and removes unused instructions
  - obfuscator: renames classes, fields, and methods using short meaningless names
- **DexGuard:**
  - code & resource protection, tries to break a number of RE tools
  - string encryption, class encryption, and dex splitting

Obfuscation checks

- Some things are very easy to spot:
  - Removed unused classes like R.java, TargetApi.java ...
  - Renamed class, fields and method names
    - with ProGuard they are renamed to 'a', 'b', 'c'...
    - with DexGuard they use single characters like `ᵒᵃ... or UTF16 characters
- dexinfo is a useful tool to spot those changes:

https://github.com/poliva/dexinfo

dexinfo

obf — vim — 121×30

```
=== dexinfo 0.1 - (c) 2012-2013 Pau Oliva Fora

[] Dex file: classes-normal.dex

[] DEX magic: 64 65 78 0A 30 33 35 00
[] DEX version: 035
[] Adler32 checksum: 0x95d40574
[] SHA1 signature: 9891025c4f7d48e9980ec5c4e9d9bed68bf3ed90

[] Number of classes in the archive: 8
[] Class 1 (SuppressLint.java): 0 direct methods, 1 virtual
methods
        virtual method 1 = value
[] Class 2 (TargetApi.java): 0 direct methods, 1 virtual met
hods
        virtual method 1 = value
[] Class 3 (BuildConfig.java): 1 direct methods, 0 virtual m
ethods
        direct method 1 = <init>
[] Class 4 (HelloAndroid.java): 1 direct methods, 1 virtual
methods
        direct method 1 = <init>
        virtual method 1 = onCreate
[] Class 5 (R.java): 1 direct methods, 0 virtual methods
        direct method 1 = <init>
[] Class 6 (R.java): 1 direct methods, 0 virtual methods
        direct method 1 = <init>
```

classes-dex.txt

```
=== dexinfo 0.1 - (c) 2012-2013 Pau Oliva Fora

[] Dex file: classes-proguard.dex

[] DEX magic: 64 65 78 0A 30 33 35 00
[] DEX version: 035
[] Adler32 checksum: 0xee669793
[] SHA1 signature: 3d82a3b498a4b1cb09d9af7d2b4afd1777fcc9e5

[] Number of classes in the archive: 1
[] Class 1 (No index): 1 direct methods, 1 virtual methods
        direct method 1 = <init>
        virtual method 1 = onCreate
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

classes-proguard-dex.txt

dexinfo

obf — vim — 121×30

```
=== dexinfo 0.1 - (c) 2012-2013 Pau Oliva Fora

[] Dex file: classes-normal.dex

[] DEX magic: 64 65 78 0A 30 33 35 00
[] DEX version: 035
[] Adler32 checksum: 0x95d40574
[] SHA1 signature: 9891025c4f7d48e9980ec5c4e9d9bed68bf3ed90

[] Number of classes in the archive: 8
[] Class 1 (SuppressLint.java): 0 direct methods, 1 virtual
methods
        virtual method 1 = value
[] Class 2 (TargetApi.java): 0 direct methods, 1 virtual met
hods
        virtual method 1 = value
[] Class 3 (BuildConfig.java): 1 direct methods, 0 virtual m
ethods
        direct method 1 = <init>
[] Class 4 (HelloAndroid.java): 1 direct methods, 1 virtual
methods
        direct method 1 = <init>
        virtual method 1 = onCreate
[] Class 5 (R.java): 1 direct methods, 0 virtual methods
        direct method 1 = <init>
[] Class 6 (R.java): 1 direct methods, 0 virtual methods
        direct method 1 = <init>
```

classes-dex.txt

```
=== dexinfo 0.1 - (c) 2012-2013 Pau Oliva Fora

[] Dex file: classes-proguard.dex

[] DEX magic: 64 65 78 0A 30 33 35 00
[] DEX version: 035
[] Adler32 checksum: 0xee669793
[] SHA1 signature: 3d82a3b498a4b1cb09d9af7d2b4afd1777fcc9e5

[] Number of classes in the archive: 1
[] Class 1 (No index): 1 direct methods, 1 virtual methods
        direct method 1 = <init>
        virtual method 1 = onCreate
```

classes-proguard-dex.txt

dexinfo verbose output

obf — vim — 121×30

```
Number of classes in the archive: 8

Class 4 (HelloAndroid.java)
    class_idx='0xa':Lcom/example/helloandroid/HelloAndroid;
    access_flags='0x1': public
    superclass_idx='0x4':Landroid/app/Activity;
    interfaces_off='0x0'
    source_file_idx='0x8':HelloAndroid.java
    annotations_off=0x0
    class_data_off=0x5b4 (1460)
    static_values_off=0x0 (0)
    0 static fields
    0 instance fields
    1 direct methods
    direct method 1 = <init>
        method_code_off=0x62c
        method_access_flags='0x10001'
        class_idx='0xa'
        proto_idx=0x1
    1 virtual methods
    virtual method 1 = onCreate
        method_code_off=0x644
        method_access_flags='0x1'
        class_idx=0xa
        proto_idx=0x5
```

1.txt [+]

```
Number of classes in the archive: 1

Class 1 (No index):
    class_idx='0x7':Lcom/example/helloandroid/HelloAndroid;
    access_flags='0x1': public
    superclass_idx='0x2':Landroid/app/Activity;
    interfaces_off='0x0'
    source_file_idx='0xffffffff'
    annotations_off=0x0
    class_data_off=0x378 (888)
    static_values_off=0x0 (0)
    0 static fields
    0 instance fields
    1 direct methods
    direct method 1 = <init>
        method_code_off=0x1b0
        method_access_flags='0x10001'
        class_idx='0x7'
        proto_idx=0x0
    1 virtual methods
    virtual method 1 = onCreate
        method_code_off=0x1c8
        method_access_flags='0x1'
        class_idx=0x7
        proto_idx=0x4
```

2.txt [+]

dexinfo verbose output

obf — vim — 121×30

```
Number of classes in the archive: 8

Class 4 (HelloAndroid.java)
    class_idx='0xa':Lcom/example/helloandroid/HelloAndroid;
    access_flags='0x1': public
    superclass_idx='0x4':Landroid/app/Activity;
    interfaces_off='0x0'
    source_file_idx='0x8':HelloAndroid.java
    annotations_off=0x0
    class_data_off=0x5b4 (1460)
    static_values_off=0x0 (0)
    0 static fields
    0 instance fields
    1 direct methods
    direct method 1 = <init>
        method_code_off=0x62c
        method_access_flags='0x10001'
        class_idx='0xa'
        proto_idx=0x1
    1 virtual methods
    virtual method 1 = onCreate
        method_code_off=0x644
        method_access_flags='0x1'
        class_idx=0xa
        proto_idx=0x5
```

1.txt [+]

```
Number of classes in the archive: 1

Class 1 (No index):
    class_idx='0x7':Lcom/example/helloandroid/HelloAndroid;
    access_flags='0x1': public
    superclass_idx='0x2':Landroid/app/Activity;
    interfaces_off='0x0'
    source_file_idx='0xffffffff'
    annotations_off=0x0
    class_data_off=0x378 (888)
    static_values_off=0x0 (0)
    0 static fields
    0 instance fields
    1 direct methods
    direct method 1 = <init>
        method_code_off=0x1b0
        method_access_flags='0x10001'
        class_idx='0x7'
        proto_idx=0x0
    1 virtual methods
    virtual method 1 = onCreate
        method_code_off=0x1c8
        method_access_flags='0x1'
        class_idx=0x7
        proto_idx=0x4
```

2.txt [+]

dexinfo verbose output

obf — vim — 121×30

```
Class 7 (a.java)
    class_idx='0x4a1':Lcom/a;
    access_flags='0x601': public  interface  abstract
    superclass_idx='0x640':Ljava/lang/Object;
    interfaces_off='0x0'
    source_file_idx='0x2b8f':a.java
    annotations_off=0x0
    class_data_off=0x20808c (2130060)
    static_values_off=0x0 (0)
    0 static fields
    0 instance fields
    0 direct methods
    3 virtual methods
    virtual method 1 = a
        method_code_off=0x0
        method_access_flags='0x401'
        class_idx=0x4a1
        proto_idx=0x437
    virtual method 2 = b
        method_code_off=0x0
        method_access_flags='0x401'
        class_idx=0x4a1
        proto_idx=0x72b
    virtual method 3 = c
        method_code_off=0x0
        method_access_flags='0x401'
        class_idx=0x4a1
        proto_idx=0x72b
proguard-dex.txt [+]
```

```
    6 direct methods
    direct method 1 = <clinit>
        method_code_off=0x7588c
        method_access_flags='0x10008'
        class_idx='0x1ad'
        proto_idx=0x437
    direct method 2 = <init>
        method_code_off=0x75b34
        method_access_flags='0x10001'
        class_idx='0x1ad'
        proto_idx=0x437
    direct method 3 = a
        method_code_off=0x75b4c
        method_access_flags='0x2000a'
        class_idx='0x1ad'
        proto_idx=0x437
    direct method 4 = b
        method_code_off=0x75be8
        method_access_flags='0x20002'
        class_idx='0x1ad'
        proto_idx=0x437
    direct method 5 = c
        method_code_off=0x75c64
        method_access_flags='0x20002'
        class_idx='0x1ad'
        proto_idx=0x437
    direct method 6 = d
        method_code_off=0x75cc0
proguard-dex.txt [+]
```

dexinfo verbose output

obf — vim — 121×30

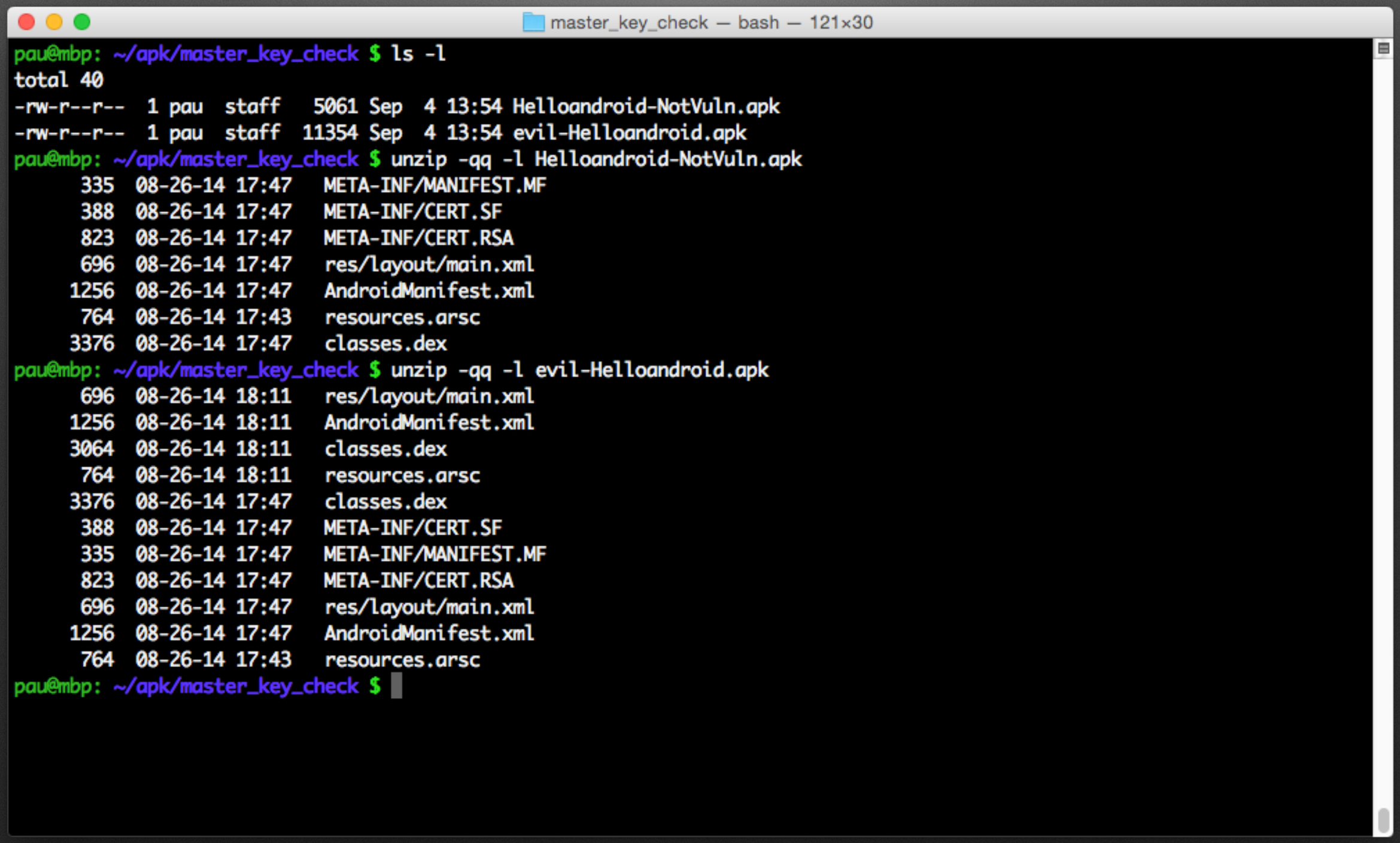
```
Class 7 (a.java)
  class_idx='0x4a1':Lcom/a;
  access_flags='0x601': public  interface  abstract
  superclass_idx='0x640':Ljava/lang/Object;
  interfaces_off='0x0'
  source_file_idx='0x2b8f':a.java
  annotations_off=0x0
  class_data_off=0x20808c (2130060)
  static_values_off=0x0 (0)
  0 static fields
  0 instance fields
  0 direct methods
  3 virtual methods
  virtual method 1 = a
      method_code_off=0x0
      method_access_flags='0x401'
      class_idx=0x4a1
      proto_idx=0x437
  virtual method 2 = b
      method_code_off=0x0
      method_access_flags='0x401'
      class_idx=0x4a1
      proto_idx=0x72b
  virtual method 3 = c
      method_code_off=0x0
      method_access_flags='0x401'
      class_idx=0x4a1
      proto_idx=0x72b
proguard-dex.txt [+]
```

```
  6 direct methods
  direct method 1 = <clinit>
      method_code_off=0x7588c
      method_access_flags='0x10008'
      class_idx='0x1ad'
      proto_idx=0x437
  direct method 2 = <init>
      method_code_off=0x75b34
      method_access_flags='0x10001'
      class_idx='0x1ad'
      proto_idx=0x437
  direct method 3 = a
      method_code_off=0x75b4c
      method_access_flags='0x2000a'
      class_idx='0x1ad'
      proto_idx=0x437
  direct method 4 = b
      method_code_off=0x75be8
      method_access_flags='0x20002'
      class_idx='0x1ad'
      proto_idx=0x437
  direct method 5 = c
      method_code_off=0x75c64
      method_access_flags='0x20002'
      class_idx='0x1ad'
      proto_idx=0x437
  direct method 6 = d
      method_code_off=0x75cc0
proguard-dex.txt [+]
```

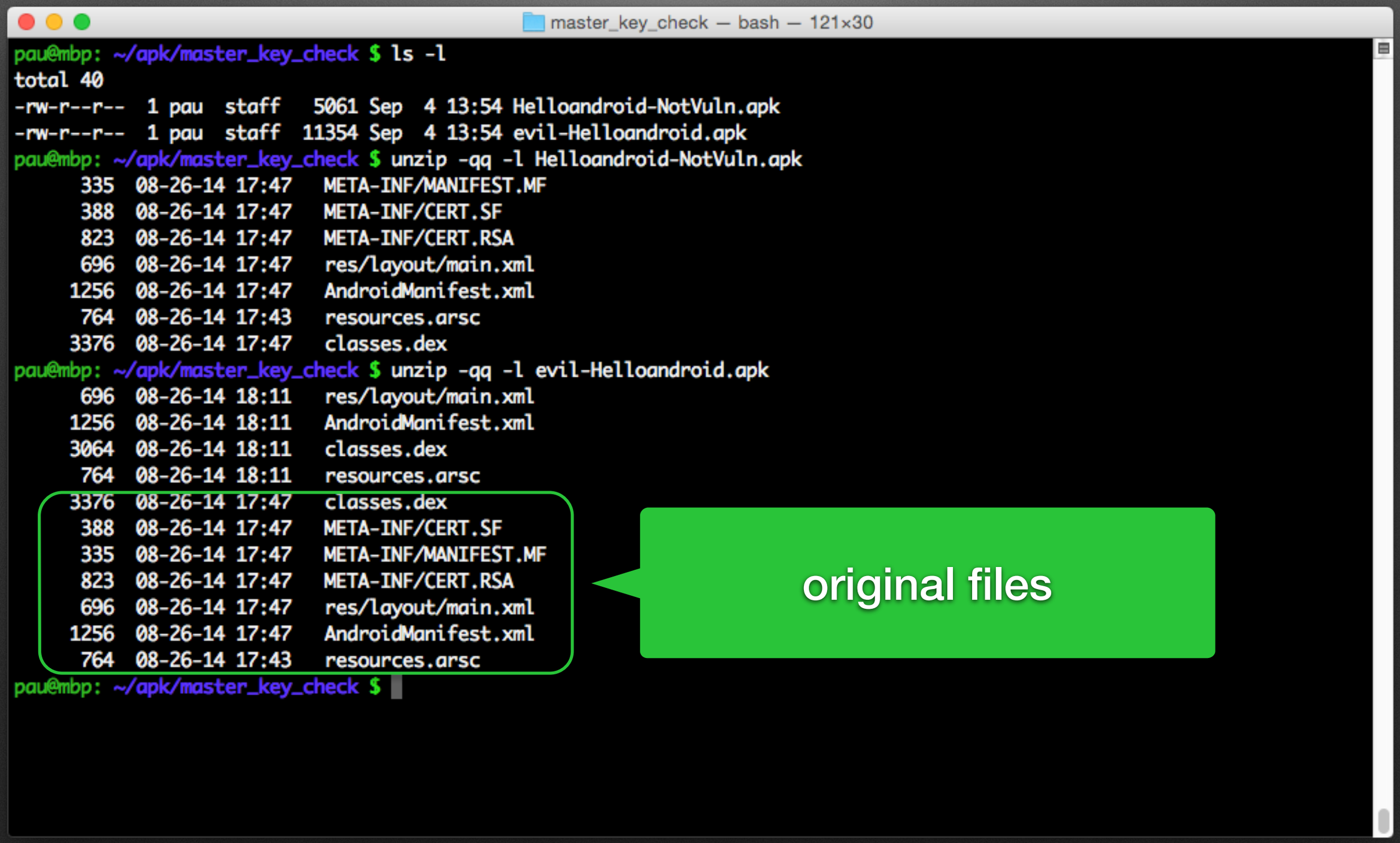
MasterKey exploit check

- Discovered by Jeff Forristal, made public on Jul 2013
- Affects all Android devices from Android 1.6 up to 4.3
- Allows duplicating entries inside an APK:
  - The hashes in META-INF folder are from the original signed files, which are checked for signature
  - The files that end up being installed on the device are the duplicated entries

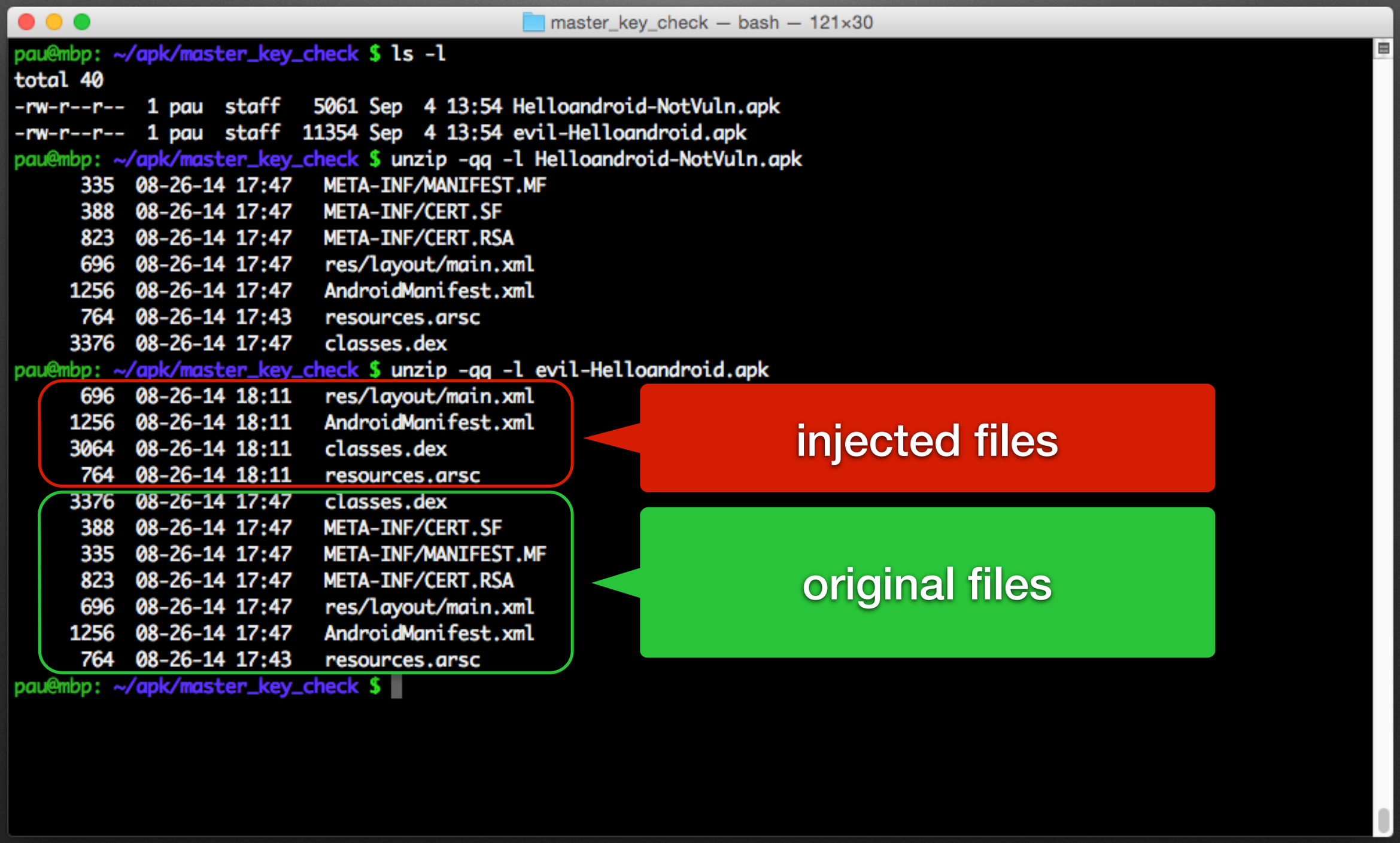
MasterKey exploit check



MasterKey exploit check



MasterKey exploit check



SecureRandom bug check

- Affects Android versions from Android 4.1 up to 4.3
- Applications using JCA (Java Cryptography Architecture) for key generation, signing, or random number generation may not receive cryptographically strong values due to improper initialization of the underlying OpenSSL PRNG.
- Allowed theft of funds to all bitcoin wallets generated using an Android APP (August 2013)

SecureRandom bug check

- Steps to check if an app is vulnerable:

1. Check if the app invokes SecureRandom()
2. Make sure the app invokes any other java.security or javax.crypto APIs
3. Check if it does invoke SetSeed() *and* has references to /dev/random or /dev/urandom

SecureRandom bug check

- Steps to check if an app is vulnerable:

1. Check if the app invokes SecureRandom()

```
strings classes.dex |grep "java.security.SecureRandom"
```

2. Make sure the app invokes any other java.security or javax.crypto APIs

3. Check if it does invoke SetSeed() *and* has references to /dev/random or /dev/urandom

SecureRandom bug check

- Steps to check if an app is vulnerable:

1. Check if the app invokes SecureRandom()

```
strings classes.dex |grep "java.security.SecureRandom"
```

2. Make sure the app invokes any other java.security or javax.crypto APIs

```
strings classes.dex |egrep "java.security|javax.crypto" \
                       |egrep -v "SecureRandom|SetSeed"
```

3. Check if it does invoke SetSeed() *and* has references to /dev/random or /dev/urandom

SecureRandom bug check

- Steps to check if an app is vulnerable:

1. Check if the app invokes SecureRandom()

```
strings classes.dex |grep "java.security.SecureRandom"
```

2. Make sure the app invokes any other java.security or javax.crypto APIs

```
strings classes.dex |egrep "java.security|javax.crypto" \
                        |egrep -v "SecureRandom|SetSeed"
```

3. Check if it does invoke SetSeed() *and* has references to /dev/random or /dev/urandom

```
strings classes.dex |grep "SetSeed"
strings classes.dex |egrep "/dev/u?random"
```

SecureRandom bug check

SecRand — vim — 121×30

```
#!/bin/bash

rm classes.dex 2>/dev/null
unzip -q $1 classes.dex || exit 1

count=$(strings classes.dex |grep "java.security.SecureRandom" |wc -l)
if [ $count -eq 0 ]; then
        echo "NOT VULN: App doesn't invoke SecureRandom" ; exit
fi

count=$(strings classes.dex |egrep "java.security|javax.crypto" |egrep -v "SecureRandom|SetSeed" |wc -l)
if [ $count -eq 0 ]; then
        echo "NOT VULN: App doesn't use JCA for critical random number generation" ; exit
fi

count=$(strings classes.dex |grep "SetSeed" |wc -l)
if [ $count -ge 1 ]; then
        count=$(strings classes.dex |egrep "/dev/u?random" |wc -l)
        if [ $count -ge 1 ]; then
                echo "NOT VULN: App properly seeds the PRNG"
        else
                echo "VULNERABLE: App does not explicitly initialize the PRNG with entropy from /dev/urandom or /dev/rand
om"
        fi
else
        echo "VULNERABLE: App does not seed the PRNG"
fi
~
"secrand.sh" 27L, 840C written
```

SecureRandom bug check

SecRand — bash — 121×30

```
pau@mbp: /tmp/SecRand $ ls -l
total 6304
-rw-r--r--  1 pau  wheel     5061 Sep  8 00:47 HelloAndroid.apk
-rw-r--r--  1 pau  wheel  1426468 Sep  7 23:46 com.coinbase.android.apk
-rw-r--r--  1 pau  wheel  1783878 Sep  7 23:46 de.schildbach.wallet.apk
-rwxr-xr-x  1 pau  wheel      840 Sep  8 00:50 secrand.sh
pau@mbp: /tmp/SecRand $ ./secrand.sh HelloAndroid.apk
NOT VULN: App doesn't invoke SecureRandom
pau@mbp: /tmp/SecRand $ ./secrand.sh de.schildbach.wallet.apk
NOT VULN: App properly seeds the PRNG
pau@mbp: /tmp/SecRand $ ./secrand.sh com.coinbase.android.apk
VULNERABLE: App does not seed the PRNG
pau@mbp: /tmp/SecRand $
```

Other useful tips

- I maintain a bunch of useful scripts for doing Android related stuff in this github repository:
  - Recompute DEX file checksum
  - Extract DEX from inside Android Runtime OAT files
  - Change MAC address on some devices...

https://github.com/poliva/random-scripts/tree/master/android

Interacting with installed APPs

Back to the basics

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

```
adb shell ls | tr '\r$' ' '
```

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

```
adb shell ls | tr '\r$' ' '
```

- Be familiar with the bash *for* loops.

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

```
adb shell ls | tr '\r$' ' '
```

- Be familiar with the bash *for* loops.

```
for f in <list>
do
    <commands>
done
```

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

```
adb shell ls | tr '\r$' ' '
```

- Be familiar with the bash *for* loops.

```
for f in <list>
do
    <commands>
done
```

```
for f in <list> ; do <commands> ; done
```

Back to the basics

- We're going to use some bash one-liners. Don't be scared! It's easy if you break them down into simple commands.
- Output of ADB shell is '\n\r' terminated, sometimes that '\r' needs to be stripped.

```
adb shell ls | tr '\r$' ' '
```

- Be familiar with the bash *for* loops.

```
for f in <list>
do
    <commands>
done
```

```
for f in <list> ; do <commands> ; done
```

```
for f in `ls` ; do echo "$f" ; done
```

Obtaining app data

backup — pau@u1404vm: ~ — bash — 122×30

```
pau@mbp: /tmp/backup $ adb shell pm list packages |tr '\r$' ' ' |grep "anydo"
package:com.anydo
pau@mbp: /tmp/backup $ adb backup -apk -obb com.anydo
Now unlock your device and confirm the backup operation.
pau@mbp: /tmp/backup $ ls -l
total 13240
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
pau@mbp: /tmp/backup $ dd if=backup.ab of=backup.zlib bs=1 skip=24
6777884+0 records in
6777884+0 records out
6777884 bytes transferred in 10.544330 secs (642799 bytes/sec)
pau@mbp: /tmp/backup $ ls -l
total 26480
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" |cat - backup.zlib | gzip -dc > backup.tar
gzip: invalid compressed data--crc error
pau@mbp: /tmp/backup $ ls -l
total 43056
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  8486912 Sep  6 12:18 backup.tar
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ tar xvf backup.tar
x apps/com.anydo/_manifest
x apps/com.anydo/a/com.anydo-1.apk
x apps/com.anydo/f/gaClientId
x apps/com.anydo/db/webviewCookiesChromiumPrivate.db
x apps/com.anydo/db/webviewCookiesChromium.db
x apps/com.anydo/db/webview.db-journal
x apps/com.anydo/db/webview.db
```

Obtaining app data

backup — pau@u1404vm: ~ — bash — 122×30

```
pau@mbp: /tmp/backup $ adb shell pm list packages |tr '\r$' ' ' |grep "anydo"
package:com.anydo
pau@mbp: /tmp/backup $ adb backup -apk -obb com.anydo
Now unlock your device and confirm the backup operation.
pau@mbp: /tmp/backup $ ls -l
total 13240
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
pau@mbp: /tmp/backup $ dd if=backup.ab of=backup.zlib bs=1 skip=24
6777884+0 records in
6777884+0 records out
6777884 bytes transferred in 10.544330 secs (642799 bytes/sec)
pau@mbp: /tmp/backup $ ls -l
total 26480
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" |cat - backup.zlib | gzip -dc > backup.tar
gzip: invalid compressed data--crc error
pau@mbp: /tmp/backup $ ls -l
total 43056
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  8486912 Sep  6 12:18 backup.tar
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ tar xvf backup.tar
x apps/com.anydo/_manifest
x apps/com.anydo/a/com.anydo-1.apk
x apps/com.anydo/f/gaClientId
x apps/com.anydo/db/webviewCookiesChromiumPrivate.db
x apps/com.anydo/db/webviewCookiesChromium.db
x apps/com.anydo/db/webview.db-journal
x apps/com.anydo/db/webview.db
```

strip android backup header

Obtaining app data

backup — pau@u1404vm: ~ — bash — 122×30

```
pau@mbp: /tmp/backup $ adb shell pm list packages |tr '\r$' ' ' |grep "anydo"
package:com.anydo
pau@mbp: /tmp/backup $ adb backup -apk -obb com.anydo
Now unlock your device and confirm the backup operation.
pau@mbp: /tmp/backup $ ls -l
total 13240
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
pau@mbp: /tmp/backup $ dd if=backup.ab of=backup.zlib bs=1 skip=24
6777884+0 records in
6777884+0 records out
6777884 bytes transferred in 10.544330 secs (642799 bytes/sec)
pau@mbp: /tmp/backup $ ls -l
total 26480
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" |cat - backup.zlib | gzip -dc > backup.tar
gzip: invalid compressed data--cr[illegible] error
pau@mbp: /tmp/[illegible]
total 43056
-rw-r-----  1 [illegible]  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  8486912 Sep  6 12:18 backup.tar
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ tar xvf backup.tar
x apps/com.anydo/_manifest
x apps/com.anydo/a/com.anydo-1.apk
x apps/com.anydo/f/gaClientId
x apps/com.anydo/db/webviewCookiesChromiumPrivate.db
x apps/com.anydo/db/webviewCookiesChromium.db
x apps/com.anydo/db/webview.db-journal
x apps/com.anydo/db/webview.db
```

strip android backup header

gzip magic

Obtaining app data

backup — pau@u1404vm: ~ — bash — 122×30

```
pau@mbp: /tmp/backup $ adb shell pm list packages |tr '\r$' ' ' |grep "anydo"
package:com.anydo
pau@mbp: /tmp/backup $ adb backup -apk -obb com.anydo
Now unlock your device and confirm the backup operation.
pau@mbp: /tmp/backup $ ls -l
total 13240
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
pau@mbp: /tmp/backup $ dd if=backup.ab of=backup.zlib bs=1 skip=24
6777884+0 records in
6777884+0 records out
6777884 bytes transferred in 10.544330 secs (642799 bytes/sec)
pau@mbp: /tmp/backup $ ls -l
total 26480
-rw-r-----  1 pau  wheel  6777908 Sep  6 12:18 backup.ab
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" |cat - backup.zlib | gzip -dc > backup.tar
gzip: invalid compressed data--cr[illegible] error
pau@mbp: /tmp/[illegible]
total 43056
-rw-r-----  1 [illegible]
-rw-r--r--  1 pau  wheel  8486912 Sep  6 12:18 backup.tar
-rw-r--r--  1 pau  wheel  6777884 Sep  6 12:18 backup.zlib
pau@mbp: /tmp/backup $ tar xvf backup.tar
x apps/com.anydo/_manifest
x apps/com.anydo/a/com.anydo-1.apk
x apps/com.anydo/f/gaClientId
x apps/com.anydo/db/webviewCookiesChromiumPrivate.db
x apps/com.anydo/db/webviewCookiesChromium.db
x apps/com.anydo/db/webview.db-journal
x apps/com.anydo/db/webview.db
```

strip android backup header

gzip magic

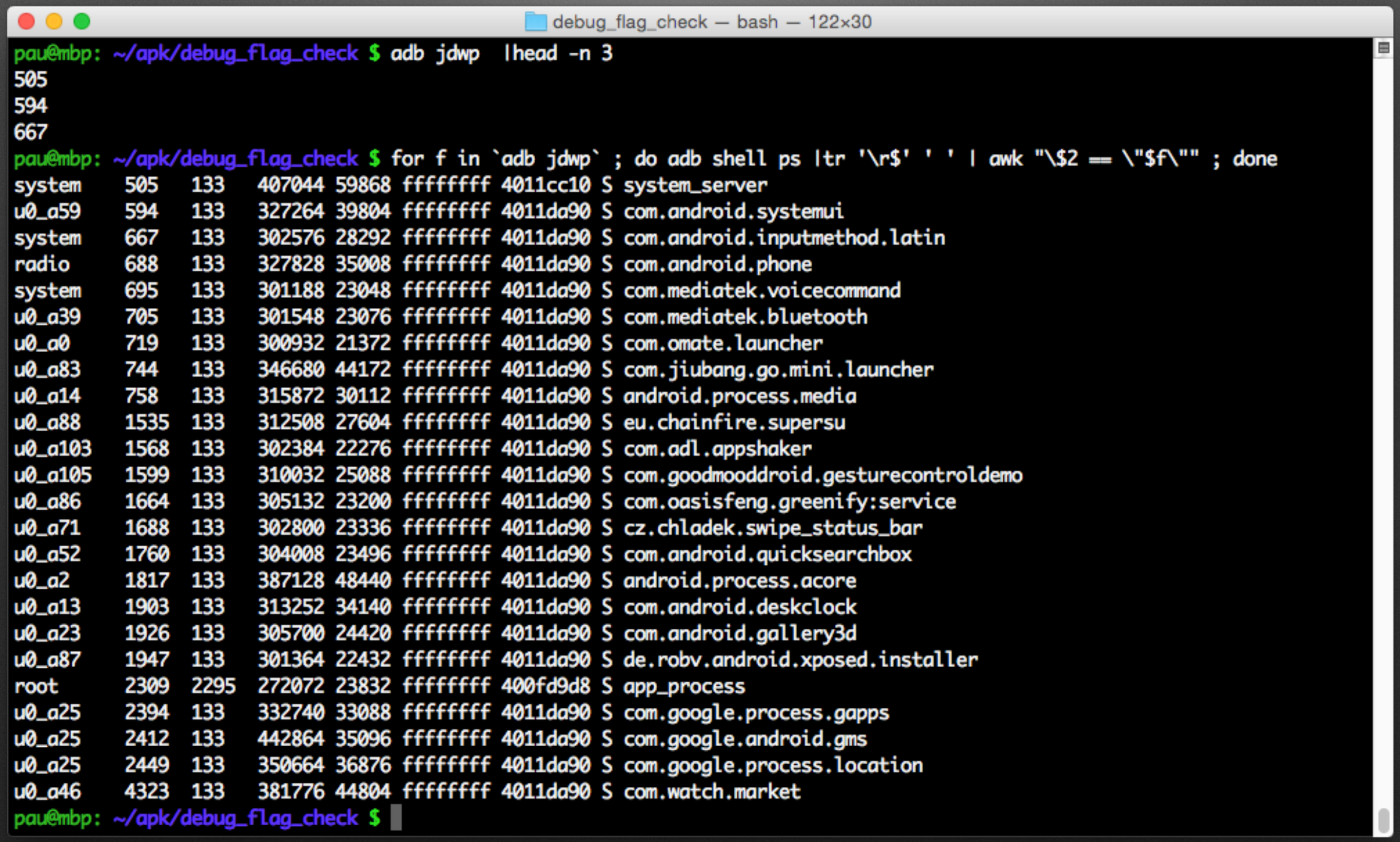
deflate compression method

Obtaining app data

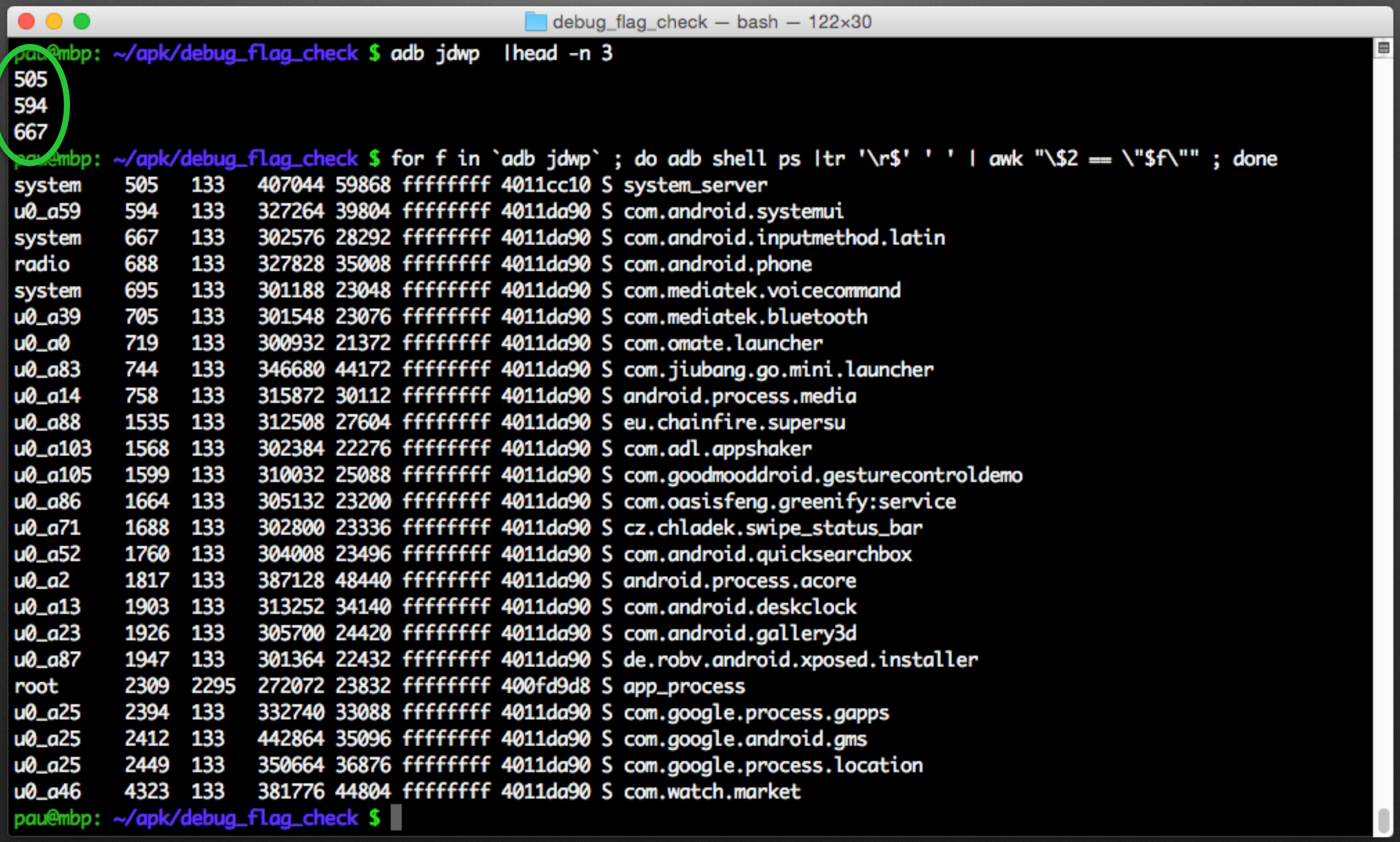
backup — pau@u1404vm: ~ — bash — 122×30

```
x apps/com.anydo/db/webview.db-journal
x apps/com.anydo/db/webview.db
x apps/com.anydo/db/data-journal
x apps/com.anydo/db/data
x apps/com.anydo/sp/user_prefs.xml
x apps/com.anydo/sp/commons_app_lifecycle.xml
x apps/com.anydo/sp/mat_log_id_install.xml
x apps/com.anydo/sp/me.kiip.sdk.xml
x apps/com.anydo/sp/mat_app_version.xml
x apps/com.anydo/sp/mat_install.xml
x apps/com.anydo/sp/mat_id.xml
x apps/com.anydo/sp/com.anydo_preferences.xml
pau@mbp: /tmp/backup $ cat apps/com.anydo/sp/user_prefs.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <boolean name="first_use_complete" value="true" />
</map>
pau@mbp: /tmp/backup $ sqlite3 apps/com.anydo/db/data .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE android_metadata (locale TEXT);
INSERT INTO "android_metadata" VALUES('en_US');
CREATE TABLE `anydo_auto_complete_cache` (`a` VARCHAR , `b` VARCHAR , `c` VARCHAR , `d` VARCHAR );
CREATE TABLE `anydo_zipped_ac_stats` (`data` BLOB , `id` INTEGER PRIMARY KEY AUTOINCREMENT );
CREATE TABLE `anydo_task_history` (`count` INTEGER , `meta_data` BLOB , `modification_time` VARCHAR , `title` VARCHAR , PR
IMARY KEY (`title`) );
INSERT INTO "anydo_task_history" VALUES(16,NULL,'2015-03-20 14:31:05.000738','call Mom');
INSERT INTO "anydo_task_history" VALUES(17,NULL,'2015-03-20 14:31:05.000902','buy Ice Cream');
INSERT INTO "anydo_task_history" VALUES(17,NULL,'2015-03-20 14:31:06.000149','go to Japan');
INSERT INTO "anydo_task_history" VALUES(1,NULL,'2015-03-20 14:30:56.000592','Pick up the kids from school');
```

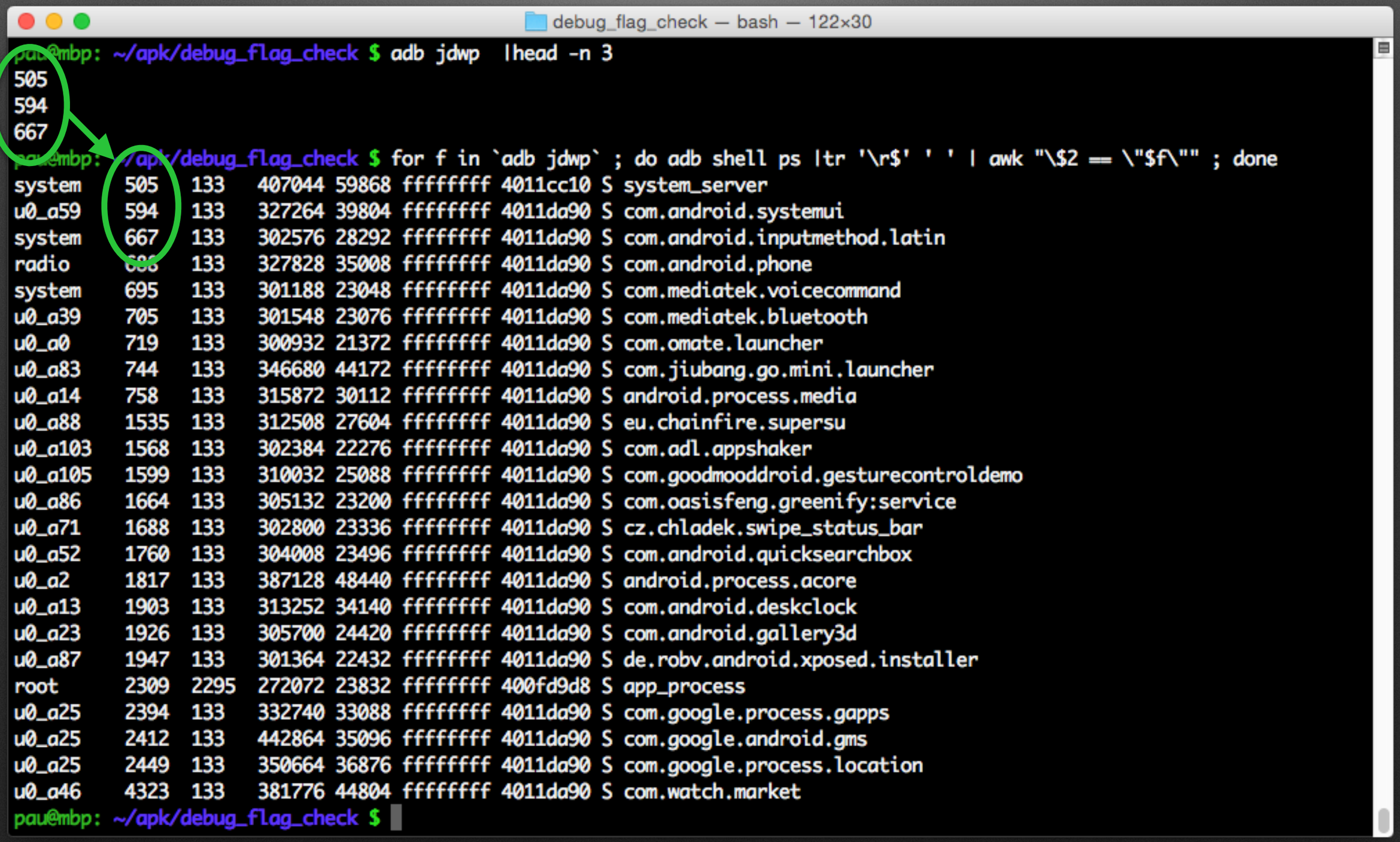
Debuggable processes



Debuggable processes



Debuggable processes



Debuggable apps

debug_flag_check — bash — 122×30

```
pau@mbp: ~/apk/debug_flag_check $ ls -l
total 32
-rw-r--r--  1 pau  staff  5051 Sep  4 13:54 HelloAndroid-debug.apk
-rw-r--r--  1 pau  staff  5061 Sep  4 13:54 HelloAndroid-release.apk
pau@mbp: ~/apk/debug_flag_check $ adb install -r HelloAndroid-debug.apk
914 KB/s (5051 bytes in 0.005s)
        pkg: /data/local/tmp/HelloAndroid-debug.apk
Success
pau@mbp: ~/apk/debug_flag_check $ adb shell pm list packages |tr '\r$' ' ' |tail
package:com.android.facelock
package:com.google.android.calendar
package:com.android.shell
package:com.android.providers.downloads
package:com.google.android.videoeditor
package:com.android.musicfx
package:com.google.android.syncadapters.contacts
package:com.google.android.apps.books
package:com.android.phasebeam
package:com.google.android.backup
pau@mbp: ~/apk/debug_flag_check $ for f in `adb shell pm list packages |tr '\r' ' ' |cut -f 2- -d ":"` ; do echo -n "$f >
" ; adb shell "run-as $f id" |tr '\r' ' ' ; done |grep -v "Package"
com.example.helloandroid > uid=10067(u0_a67) gid=10067(u0_a67) context=u:r:untrusted_app:s0
pau@mbp: ~/apk/debug_flag_check $
```

Questions?

Thank you!